Verliesloze beeldcompressie en statistische voorspelling aan de hand van contextmodellering

Koen Denecker

Promotor: Prof. dr. I. Lemahieu
Prof. dr. ir. W. Philips

Proefschrift ingediend tot het behalen van de graad van Doctor in de Toegepaste Wetenschappen: Computerwetenschappen

Vakgroep Elektronica en Informatiesystemen
Voorzitter: Prof. dr. ir. J. Van Campenhout
Faculteit Toegepaste Wetenschappen
Academiejaar 2002 – 2003

Dankwoord

Dit werk zou niet mogelijk geweest zijn zonder de steun, de medewerking en het geduld van een groot aantal personen. Ik zou dan ook graag deze gelegenheid aanwenden om hen mijn oprechte dank te betuigen.

In de eerste plaats gaat mijn dank uit naar mijn promotoren Ignace Lemahieu en Wilfried Philips. Ze hebben mij de kans geboden in hun onderzoeksgroep aan de slag te gaan en er mijn onderzoek uit te voeren. Bovendien hebben ze mij te allen tijde onvoorwaardelijk gesteund op alle mogelijke vlakken en zijn ze verantwoordelijk voor het creëren van een omgeving waar het mogelijk is om onderzoek te doen op een internationaal niveau.

Verder wens ik de vakgroepvoorzitter Jan Van Campenhout te bedanken, niet alleen voor de steun die hij mij geboden heeft, maar ook voor de spreekwoordelijke vonk die hij meer dan eens liet overspringen.

Een aantal resultaten in dit werk heb ik te danken aan de actieve bijdrage van en geanimeerde dicussies met twee ex-collega's: Steven Van Assche en Dimitri Van De Ville. De firma Barco Graphics NV (recent hernoemd tot Esko-Graphics NV) en Hans De Stecker in het bijzonder wens ik te bedanken voor hun bijdrage tot het aflijnen van de probleemstelling en hun actieve steun tijdens het onderzoek. Verder wens ik ook Rudi Vander Vennet en John Crombez van de vakgroep Financiële economie te bedanken voor hun bijdrage en ondersteuning in het onderzoek naar financiële modellering en risicoanalyse.

Dit proefschrift zou er niet gekomen zijn zonder de niet aflatende ruggensteun van de volgende personen binnen en buiten de vakgroep en ik wens hun daarvoor te bedanken: Peter de Neve, Rik Van de Walle, Marnik Brunfaut, Wim Meeus, Koen De Bosschere, Frederik Habils, Peter Schelkens, Ronny Blomme en Frank Sommen.

Een aantal mensen binnen de vakgroep verdienen terecht een teken van dank omwille van hun bijdrage tot de stimulerende en eeuwig positieve sfeer. Er zijn niet alleen mijn ex-bureaucollega's en vrienden Peter De Neve, Didier Bouden en Jeroen Van Overloop, maar ook Rita Breems kan niet genoeg bedankt worden voor haar administratieve en persoonlijke steun.

Voor het kritisch nalezen van dit werk kon ik rekenen op de geduldige inzet

van Wilfried Philips, Ignace Lemahieu en Bart Denecker.

Verder wens ik de talloze professoren die bijgedragen hebben tot mijn wetenschappelijke vorming te bedanken, evenals de internationale onderzoekers met wie ik vruchtbare discussies gevoerd heb, de onderzoekers die hun software vrij ter beschikking stelden en de anonieme recensenten die meer dan eens terechte maar constructieve kritiek gaven over mijn onderzoek.

Tot slot gaat mijn gemeende dank uit naar mijn familie en een aantal vrienden uit de persoonlijke omgeving die nooit twijfelden aan het welslagen van deze missie.

Dit doctoraat is mede tot stand gekomen dankzij een mandaat als aspirant bij het Fonds voor Wetenschappelijk Onderzoek – Vlaanderen.

Koen Denecker
21 oktober 2002

Inhoud

Hoofdstuk 1

Inleiding

1.1 Verliesloze compressie

Kort na het ontstaan van de eerste computers ontstond de vraag naar een machinevoorstelling van gegevens die op deze computers zouden verwerkt worden. Initieel was het belangrijk om de gegevens betrouwbaar te kunnen voorstellen zonder de algemeenheid van de eigenlijke voorstelling in het gedrang te brengen. Maar al vroeg werd het belang erkend van de efficiëntie van deze voorstelling [221–223]. Dit vormt de geboorte van de datacompressie.

De wetenschap van de datacompressie stelt zich tot doel om een equivalente voorstelling van een object te genereren die korter is dan de oorspronkelijke voorstelling. Het *telargument* toont aan dat dit niet mogelijk is voor alle objecten. Er bestaan namelijk minder objecten met een kortere voorstelling dan er objecten bestaan met de oorspronkelijke voorstelling, dus moet het comprimeren van sommige objecten onvermijdelijk resulteren in een langere voorstelling. Een goede compressietechniek slaagt er in de typische van de atypische te onderscheiden en voor elk van de typische objecten een voorstelling van minimale lengte te genereren.

Het gebruik van datacompressie heeft een groot aantal voordelen. Zo laat het toe om gegevens efficiënter op te slaan, wat resulteert in kostenbesparing en wat tevens nieuwe toepassingen creëert zoals digitale archivering. Daarenboven is het mogelijk om gegevens veel sneller en efficiënter door te sturen. Een alledaagse toepassing zoals het faxen van documenten zou tot twintig keer langer duren zonder compressie. En het Internet zou nooit zo snel populair geworden zijn zonder de compressie van multimedia-objecten zoals beelden en geluid. Kortom, datacompressie is een essentiële bouwsteen die toelaat de digitalisering van onze omgeving versneld door te voeren.

Maar datacompressie heeft ook een aantal nadelen: het is een extra ver-

werkingsstap die tijd vraagt en die bovendien een reken- en geheugenkost met zich meebrengt. Daarenboven leidt het tot een toename van de globale complexiteit van het verwerkingsproces en bijgevolg zijn standaarden essentieel om compatibiliteitsproblemen te vermijden. Het is de uitdaging om voor elke potentiële toepassing de voordelen en de nadelen van compressie af te wegen en een gepaste keuze te maken temidden van het gamma van beschikbare technieken.

Door de jaren heen zijn de technieken voor datacompressie gaandeweg verfijnd en geoptimaliseerd. Enerzijds kan de aard van de objecten meer variëren dan ooit, wat aanleiding geeft tot compressietechnieken die geoptimaliseerd zijn voor bepaalde datatypes. Zo bestaan er momenteel technieken voor teksten, voor stilstaande beelden, voor bewegende beelden, voor zuiver numerieke gegevens, voor DNA, enzoverder. Anderzijds is de computerkracht dermate toegenomen dat tegenwoordig heel geavanceerde rekenmodellen probleemloos kunnen geïmplementeerd worden op alledaagse toestellen. Het is momenteel mogelijk om kleurenbeelden te decomprimeren op een mobiele telefoon of gecomprimeerde muziek aan CD-kwaliteit af te spelen op een draagbare MP3-speler.

De belangrijkste eigenschap van een compressietechniek is ongetwijfeld het effect op het object na toepassing van compressie en decompressie. Blijft de binaire voorstelling van het object ongewijzigd, dan spreken we van *verliesloze compressie* (*E.* lossless compression). Dit is de meest gebruikte techiek voor teksten en pure machinegegevens waar wijzigingen ontoelaatbaar zijn. Maar ook voor een aantal beeldklassen zoals medische beelden en beelden uit de drukvoorbereidingsindustrie verdient deze aanpak de voorkeur. Treden er na decompressie kleine en nagenoeg onmerkbare wijzigingen op, dan gebruiken we de term *verlieshebbende compressie* (*E.* lossy compression). Deze aanpak verdient de voorkeur voor veel multimedia-toepassingen voor eindgebruikers. De impact van de wijzigingen is hier beperkt omdat de oorspronkelijke voorstelling sowieso niet exact is als gevolg van het analoge ontstaansproces. De keuze tussen verliesloze en verlieshebbende compressie is afhankelijk van de toepassing en betekent een afwegen tussen hogere compressie enerzijds en behoud van informatie en detail anderzijds.

1.2 Probleemstelling

Het onderzoek in dit werk spitst zich toe op de verliesloze compressie van hogeresolutiebeelden uit de drukvoorbereidingsindustrie. We staan niet alleen stil bij de efficiëntie van state-of-the-art technieken, maar stellen ook een aantal aanpassingen en uitbreidingen voor die geoptimaliseerd zijn voor

halftoonbeelden. We tonen aan dat deze optimalisaties efficiënt kunnen geïmplementeerd worden en vrij universeel zijn. Ze kunnen met succes uitgebreid worden naar andere objecten en toepassingen variërend van contone residubeelden tot risicoanalyse van financiële gegevens.

De cruciale vraag in dit werk is gericht op de modelvorming, die het hart vormt van elk compressieschema. De andere aspecten zoals de eigenlijke entropiecodering zijn voldoende gekend en begrepen, al worden ze wel beschreven. De zoektocht naar de optimale modelvorming van waargenomen gegevens laat niet alleen efficiënte compressie toe, het vormt ook het startpunt voor een aantal nieuwe toepassingen zoals financiële modellering, optische karakterherkenning, spraakherkenning, fraudedetectie, encryptie, optimale indexering voor zoekopdrachten, spellingscontrole, vertaalmodellen, enzoverder. Bij wijze van voorbeeld gaan we in dit werk dieper in op de modellering van financiële gegevens met risicoanalyse als specifieke doelstelling.

Deze parallellen vormen precies de schoonheid die ervoor zorgt dat de “datacompressie als wetenschap” de “datacompressie als ambacht” overstijgt. Datacompressie als wetenschap is de zoektocht naar de essentie van het waargenomen gedrag. Het zo nauwkeurig mogelijk inschatten van deze essentie opent de deuren naar een waaier van nieuwe mogelijkheden, die gezamenlijk de oorspronkelijke doelstelling van datacompressie overstijgen. Datacompressie als ambacht daarentegen is uitsluitend gericht op maximale compressie aan minimale geheugen- en rekenkost. Deze aanpak is helaas onderworpen aan fundamentele grenzen net zoals de temperatuur niet onder het absolute nulpunt kan. Hoe dichter datacompressie deze fundamentele grens nadert, hoe moeilijker het is om de bekomen compressie nog te doen toenemen.

1.3 Overzicht van het proefschrift

Dit proefschrift is als volgt samengesteld. Hoofdstuk 2 beschrijft een aantal basisconcepten uit de datacompressie en de informatietheorie, zoals het algemeen compressieschema, de classificaties van de modeltypes en de begrippen entropie en complexiteit.

Hoofdstuk 3 handelt uitvoerig over entropiecodes en hun implementatie. Entropiecodering vormt het bouwblok dat in elke compressietechniek instaat voor de eigenlijke generatie van de equivalente maar kortere voorstelling.

Vervolgens gaan we dieper in op de compressie van tekst en beelden in het algemeen en van beelden uit de drukvoorbereidingsindustrie in het bijzonder. In hoofdstuk 4 geven we een gedetailleerd overzicht van de state of the art van verliesloze beeldcompressie. De bestaande technieken worden opgesplitst volgens hun toepassingsgebied en geanalyseerd met aandacht voor de

onderliggende modelvorming. Voor de testbeelden voorgesteld in appendix A geven we de experimentele resultaten weer. Daarnaast komen nog een aantal specifieke optimalisaties aan bod voor de drukvoorbereidingsindustrie.

Hoofdstuk 5 stelt voor hoe contextmodellering, momenteel de basis voor de beste technieken, geoptimaliseerd kan worden voor halftoonbeelden. Een efficiënte implementatie in software en in hardware zorgt ervoor dat deze optimalisaties niet ten koste gaan van de verwerkingssnelheid. Is snelheid niet het belangrijkste criterium, dan kan het model nog verder verfijnd worden naar een contextmodel van variabele orde. Verder is het mogelijk de voorgestelde optimalisaties te gebruiken voor residucodering van grijswaarden- en kleurenbeelden, wat aanleiding geeft tot een universele en efficiënte aanpak.

Het aanwenden van een contextmodel als basistechniek voor de modelvorming beperkt zich niet tot verliesloze compressie. Hoofdstuk 6 beschrijft welke aanpassingen nodig zijn om contextmodellering te gebruiken voor de voorspelling van de value-at-risk, een statistische maat voor het marktrisico in de financiële wereld. Experimenten op historische waarnemingen tonen het anticiperend potentieel aan van deze aanpak.

Tot slot vatten we de belangrijkste resultaten samen en geven we een overzicht van de eigen bijdragen in hoofdstuk 7.

1.4 Publicaties

Het onderzoek voorgesteld in dit proefschrift heeft geresulteerd in vijf publicaties in internationale tijdschriften waarvan drie als eerste auteur [55, 57, 64, 178, 251]. Een deel van de resultaten zijn eveneens voorgesteld in eenentwintig bijdragen op internationale conferenties waarvan vijftien als eerste auteur [47,49–52,54,56,58–63,65,66,176,249,250,252,254,255]. In het kader van het onderzoek naar beeldcompressie en beeldverwerking heeft voortzetting van vroeger onderzoek, verdere medewerking binnen de onderzoeksgroep en begeleiding van thesisstudenten geleid tot nog vijf publicaties in een internationaal tijdschrift waarvan twee als eerste auteur en nog eenentwintig bijdragen op internationale conferenties. Appendix C geeft een opsomming van alle publicaties die verschenen zijn gedurende de loop van dit onderzoek.

Hoofdstuk 2

Beeldcompressie en basisconcepten uit de informatietheorie

2.1 Inleiding

Nagenoeg alle technieken voor compressie zijn opgebouwd volgens eenzelfde basisaanpak. Centraal in deze aanpak staat de eigenlijke modelvorming die zich bezig houdt met het definiëren van een typisch bronobject en het berekenen van de probabiliteit van een waargenomen bronobject binnen de klasse van de typische bronobjecten. Binnen deze algemene aanpak onderscheiden de bestaande technieken zich volgens een aantal criteria.

Al wordt de uiteindelijke efficiëntie van een compressietechniek bepaald door de specifieke aanpak en implementatie, toch loont het de moeite om vooraf even stil te staan bij een aantal eigenschappen die gemeenschappelijk zijn voor alle compressietechnieken. Onafhankelijk van de techniek zijn er een aantal belangrijke grootheden zoals de entropie die een direct verband hebben met de haalbare efficiëntie. Verder voldoen alle technieken ook aan een aantal essentiële wetmatigheden. Deze beschrijving geeft inzicht in de fundamenten van de datacompressie en toont de verbanden met andere theorieën zoals de stochastiek, de informatietheorie en de complexiteitstheorie.

In dit hoofdstuk ligt de nadruk vooral op de uiterlijke kenmerken van de compressietechnieken en het globaal kader eromheen. De invalshoek is eerder beschrijvend en maakt gebruik van concepten en definities die in de verdere hoofdstukken gedetailleerd aan bod komen. Er komen abstracte compressietechnieken aan bod die een louter theoretisch belang hebben. Paragraaf 2.2

beschrijft het algemeen compressieschema vanop een hoog niveau en gaat dieper in op de redundantie, de verschillende modeltypes en de onderscheidende eigenschappen van een beeldcompressietechniek. Vervolgens komen in paragraaf 2.3 een aantal basisgrootheden en -concepten uit de informatietheorie aan bod, evenals hun verband met de datacompressie. Paragraaf 2.4 beschrijft een aantal parallellen met de complexiteitstheorie. Tenslotte vat paragraaf 2.5 de belangrijkste concepten en parallellen nog eens samen.

2.2 Beeldcompressie

2.2.1 Algemeen compressieschema

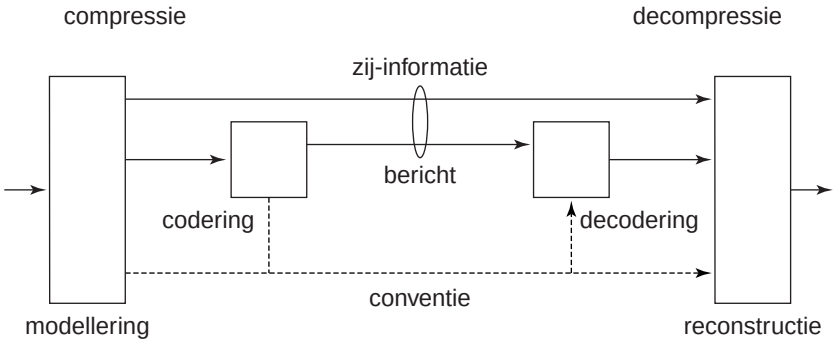
Nagenoeg alle hedendaagse compressietechnieken zijn opgesteld volgens het stramien van het *algemeen compressieschema*. Figuur 2.1 stelt dit schema voor, vertrekkend van de oospronkelijke voorstelling van het object over de gecomprimeerde voorspelling en tot de gereconstrueerde voorstelling. Het transmissiekanaal wordt hier foutvrij verondersteld.

Het linkergedeelte stelt het compressiegedeelte voor en bestaat uit enerzijds een decorrelatieblok dat de redundantie verwijdert en anderzijds een blok dat voor de entropiecodering instaat. Het rechtergedeelte stelt het decompressiegedeelte voor bestaande uit de inverse blokken van het compressiegedeelte. Voor elk bronobject is de gecomprimeerde voorstelling samengesteld uit een bericht en de zij-informatie. Het *bericht* is het resultaat van de entropiecodering en stelt de eigenlijke gegevens voor. De *zij-informatie* bevat alle ongecomprimeerde objectafhankelijke informatie die noodzakelijk is voor de reconstructie. Tenslotte is er ook een hoeveelheid informatie die eenmalig moet doorgestuurd worden en hier de *conventie* genoemd wordt. Deze beschrijft onder andere het volledige gedrag van het decompressiegedeelte en kan bijvoorbeeld de vorm van een gepubliceerde standaard of een softwaremodule aannemen. Omdat deze constant is en onafhankelijk van het bronobject wordt deze informatie doorgaans niet in rekening gebracht voor het evalueren van een compressietechniek.

Het belangrijkste evaluatiecriterium is de reductie in lengte van de gecomprimeerde voorstelling ten opzichte van de oorspronkelijke voorstelling. Numeriek leidt dit tot de definitie van de *compressieverhouding* (*E.* compression ratio)

$$c_r = \frac{l(\text{bronobject})}{l(\text{bericht}) + l(\text{zij-informatie})},$$

waarbij $l(\cdot)$ de lengte van de voorstelling uitdrukt in bits. De compressieverhouding is dimensieloos en hoger dan 1 indien de compressie succesvol



Figuur 2.1: Het algemeen compressieschema bestaande uit de compressie (links) en de decompressie (rechts). De uitgewisselde informatie per bronobject bestaat uit ongecomprimeerde zij-informatie en het entropiegecodeerd bericht. De conventie is tijds- en objectonafhankelijk en wordt daarom eenmalig doorgestuurd.

verloopt. Indien de oorspronkelijke voorstelling van het object bestaat uit afzonderlijke individuele elementen van zelfde lengte zoals bijvoorbeeld letters of pixels, wordt vaak als alternatief criterium de *bitdiepte* (*E.* bit depth, bit rate) gebruikt, gedefinieerd als

$$r_b = \frac{l(\text{bericht}) + l(\text{zij-informatie})}{c(\text{elementen})},$$

waarbij $c(\cdot)$ het aantal individuele elementen voorstelt, dus $c(\text{elementen}) = l(\text{bronobject})/l(\text{element})$. De eenheid is afhankelijk van de aard van het bronobject en is uitgedrukt in “bpc” (bits per karakter) of “bps” (bits per symbool) voor tekstcompressie en in “bpp” (bits per pixel) voor beeldcompressie. In algemene termen zijn beide grootheden een maat voor de *efficiëntie* (*E.* performance) van de techniek. Naast deze twee voor de hand liggende definities komen nog een aantal andere definities en termen aan bod in de literatuur [272].

Voor het berekenen van de gemiddelde efficiëntie over een aantal n bronobjecten gaat de voorkeur uit naar het ongewogen rekenkundig gemiddelde van de individuele bitdieptes, dus $\bar{r_b} = \sum_i r_b^{(i)}/n$. Dit is equivalent aan het kiezen van het harmonisch gemiddelde van de compressieverhoudingen $n/\sum_i (1/c_r^{(i)})$ als gemiddelde compressieverhouding over een aantal bronobjecten. Volgens deze benadering speelt de absolute grootte van de bronobjecten geen rol.

2.2.2 Redundantie en decorrelatie in beelden

Compressie kan slechts winst bekomen indien er *redundantie* aanwezig is in de beelden en indien de techniek er in slaagt om deze redundantie te detecteren

en vervolgens uit te buiten. Deze laatste stap wordt *decorrelatie* genoemd en maakt doorgaans gebruik van een wiskundige transformatie die de oorspronkelijke pixelwaarden omzet in nieuwe pixelwaarden die min of meer als onafhankelijk gedistribueerd beschouwd worden.

Digitale beelden komen voor in vele gedaanten. In dit werk ligt de nadruk op gedigitaliseerde beelden van natuurlijke scènes voor de drukvoorbereidingsindustrie. Naast de contone representatie in één of meer kleuren kennen ze een equivalente tweeniveauvoorstelling als halftoonbeeld voor drukdoeleinden. Andere beeldklassen zijn de vectorbeelden, de tomografische beelden, de paletbeelden en de meerdimensionale beelden, maar deze komen hier niet expliciet aan bod.

De redundantie van een digitaal beeld kan zich op verschillende manieren uiten en kan op verschillende manieren gedefinieerd worden. Hierbij zijn de statistische en de subjectieve redundantie de belangrijkste [89]. De statistische redundantie is gebaseerd op de hogere frequentie van optreden van sommige elementen in verhouding tot andere elementen. Deze elementen kunnen zo klein zijn als een pixel, maar ook groter zoals een visueel object dat meerdere keren voorkomt, tot een terugkerend patroon dat het hele beeld beslaat. De subjectieve redundantie of *irrelevantie* daarentegen vindt zijn oorsprong in het feit dat het *menselijk visueel systeem* (*E.* human visual system) of HVS kleine afwijkingen maskeert in de visuele perceptie van beelden [106, 285]. Voor verliesloze compressie komt enkel de statistische redundantie in aanmerking.

De beelddecorrelatie voert een wiskundige transformatie uit op de beelddata waardoor beide types redundantie expliciet verwijderd kunnen worden. Deze decorrelatie staat in nauw verband met de volgorde waarin de beeldgegevens verwerkt worden en de daarmee samengaande data-afhankelijkheid. Specifiek komen de volgende vier verwerkingsschema's voor:

- **dimensiereductie**: Een scansequentie induceert een ordening van de pixels in een tweedimensionaal beeld overeenkomstig hun positie en stelt de beeldinformatie voor als een eendimensionale vector. De *rasterscan* is de meest voorkomende vorm: hierbij worden de pixels lijn na lijn verwerkt van links naar rechts en de lijnen zelf van boven naar onder. Merk op dat de rasterscan kleine afwijkingen invoert omdat de eerste pixel van een nieuwe lijn strikt beschouwd geen buur is van de laatste pixel van de vorige lijn.

- **segmentatie**: Een algoritme splitst het beeld op in gescheiden segmenten en elk van deze segmenten wordt apart behandeld. De segmenten kunnen van een constante vorm zijn (bijvoorbeeld vierkante blokken) of van een variabele vorm (bijvoorbeeld gebaseerd op de beeldinhoud).

- **multiresolutie**: Opeenvolgende herschalingen transformeren het oorspronkelijke beeld in meerdere lageresolutiekopieën die volgens toenemende resolutie gecomprimeerd worden.

- **structuur**: Een optimalisatietechniek zet het beeld (bijvoorbeeld een matrix van gehele getallen) om in een beeldafhankelijke maar minimale datastructuur (bijvoorbeeld een boomstructuur). Het verwijderen van de redundantie vindt plaats doordat de beschrijvingslengte van de datastructuur gereduceerd wordt.

Elk van deze geïnduceerde data-afhankelijkheidstypes kan aangewend worden om zowel de statistische als de subjectieve redundantie uit te buiten.

2.2.3 Beeldmodellen

Na decorrelatie is de statistische redundantie expliciet aanwezig in het gedecorreleerd beeld of in een equivalente datastructuur. Een aantal statistische modellen komen vervolgens in aanmerking om de waargenomen statistieken te kwantificeren. Er treden veel parallellen en overlappende eigenschappen op tussen deze modellen. Het *model* zelf wordt vaak gedefinieerd als de samenstelling van de structuur en de statistieken van de waarneming [136].

Het basismodel voor de gedecorreleerde waarnemingen is dat van de *discrete geheugenloze bron* (*E.* discrete memoryless source) of DMS. De gedecorreleerde pixelwaarden stellen onafhankelijke waarnemingen voor van een tijdsinvariante toevalsgrootheid. De entropiecoder codeert elk van deze uitkomsten met een lengte die de informatie van de uitkomst benadert.

De wisselwerking tussen onafhankelijke tijdsinvariante probabiliteitsschattingen en de entropiecoder heeft zijn beperkingen. Een oplossing hiervoor ligt in *alfabetuitbreiding* (*E.* alphabet extension) waarbij een aantal waarnemingen gegroepeerd worden en vervolgens als nieuwe toevalsgrootheid optreden met een eigen geassocieerde probabiliteitsschatting. Het aantal waarnemingen dat hierbij gecombineerd wordt kan vast of veranderlijk zijn. Een bijkomend voordeel van deze aanpak is dat remanente redundantie, dit is redundantie die niet volledig verwijderd is tijdens het decorrelatieproces, verder kan uitgebuit worden.

Een volgende logische uitbreiding in het verder uitbuiten van de remanente redundantie vormt het *contextmodel*. Voor het schatten van de probabiliteit van de eerstvolgende waarneming wordt een *context* gedefinieerd en enkel de historische waarnemingen horend bij een identieke context worden in rekening gebracht. De context kan bijvoorbeeld eenvoudigweg de combinatie zijn van de pixelwaarden links en boven de huidige pixel. Contextmodellen zijn heel

efficiënt en universeel en bovendien laten ze een vrij snelle implementatie toe met beperkte en controleerbare geheugenkost.

Het *probabilistische eindigetoestandsmodel* (*E.* probabilistic finite-state model) of PFSM bestaat uit een eindig aantal *toestanden* (*E.* states) en overgangswaarschijnlijkheden geassocieerd met de transities tussen deze toestanden. Met elke transitie is een uniek label geassocieerd dat overeenstemt met een nieuwe waarneming. De nieuwe toestand wordt volledig bepaald door de huidige toestand en de nieuwe waarneming. Merk op dat de eindigheid van het model slaat op het aantal toestanden en niet op de lengte van het object. De toestanden kunnen rechtstreeks volgen uit de waargenomen omgeving en in deze situatie is het equivalent aan een contextmodel. In zijn basisgedaante voor tekstcompressie is een contextmodel dus een PFSM maar niet noodzakelijk omgekeerd. De toestanden kunnen ook abstracte toestanden voorstellen zonder een directe relatie met de pixelwaarden. Om de huidige toestand van het systeem te kennen kan het in dit geval noodzakelijk zijn de volledige sequentie in het verleden te beschouwen en deze kan arbitrair lang worden. Het genereren en beheren van deze abstracte toestanden is niet triviaal. Een concreet voorbeeld van een PFSM dat niet als contextmodel kan beschreven worden is het model waar de probabiliteit van de pixelwaarde expliciet afhankelijk is van de pixelindex.

Een PFSM wordt ook wel een *markovmodel* genoemd, waarbij de eigenschap *markoviaans* er op wijst dat de overgangsprobabiliteiten tussen toestanden enkel afhankelijk zijn van de huidige toestand en niet van een verder verleden. Dit vormt vaak een bron van verwarring, want het kan zijn dat het onbegrensde verleden moet gekend zijn om de huidige toestand te bepalen. Toch vormt dit geen contradictie. In de literatuur blijkt het begrip “markovmodel” dan ook een heel flexibele interpretatie te kennen. Een contextmodel kan dan ook geïnterpreteerd worden als een eenvoudig markovmodel met benaderde toestanden, de contexten stellen immers niet de eigenlijke toestanden voor maar een praktische benadering ervan.

Daar waar een enkelvoudig model niet volstaat om het waargenomen gedrag te beschrijven kan gebruik gemaakt worden van een *samengesteldebronmodel* (*E.* composite source model). Dit stelt een samenstelling voor van atomaire bronnen, waarbij elke bron door een ander individueel bronmodel beschreven wordt. Een probabilistische schakel aggregeert deze bronnen tot een samengesteld model.

Ergodiciteit is een eigenschap die nauw verband houdt met het convergerend gedrag van een probabilistisch model. Concreet is een model *ergodisch* (*E.* ergodic) indien elke sequentie die volgt uit het model volledig representatief wordt voor het model naarmate de sequentie langer en langer wordt. Dit

betekent dat één sequentie volstaat om het onderliggende model volledig te beschrijven, op voorwaarde dat die sequentie voldoende lang is en met waarschijnlijkheid 1. Samen met eindigheid en stationariteit is ergodiciteit dikwijls een eigenschap die ondersteld moet worden om tot een werkende aanpak te komen.

2.2.4 Onderscheidende eigenschappen

De laatste decennia werden een bijzonder groot aantal beeldcompressietechnieken voorgesteld in de literatuur. De differentiatie van deze technieken kan gebeuren volgens een aantal criteria:

- **verlies**: Zoals eerder beschreven kan een techniek verliesloos of verlieshebbend zijn. Daartussen zit het begrip *quasi-verliesloos*, waarbij de individuele pixelafwijking aan een vooropgesteld criterium moet voldoen. Een subklasse van de verlieshebbende technieken is die van de gecontroleerd verlieshebbende, waarbij de globale beeldfout moet voldoen aan een vooropgestelde visuele maat.
- **efficiëntie**: De efficiëntie is een algemene term voor de behaalde compressieverhouding of bitdiepte en is een maat voor de winst in opslagruimte of transmissiesnelheid.
- **kost**: De kost drukt de vereiste processortijd of geheugenkost uit, of ook wel het gebruik van om het even welk ander bronmiddel dat beperkt beschikbaar is.
- **symmetrie**: Elk kostaspect kan symmetrisch of asymmetrisch zijn, afhankelijk van het verschil tussen de vereiste hoeveelheid voor respectievelijk het coderen en het decoderen.
- **progressiviteit**: De progressiviteit drukt het visueel reconstructieverloop uit als functie van de reconstructietijd. Een progressieve techniek genereert bij decompressie heel vlug een voorlopige grove benadering van het beeld en voegt vervolgens detail toe. De voorlopige benadering kan aan de gebruiker worden getoond.
- **data-afhankelijkheid**: De data-afhankelijkheid beschrijft in welke mat het mogelijk is om willekeurige beeldgebieden te reconstrueren zonder eerst de rest van het beeld op te bouwen.
- **vertraging**: Voor een gepijplijnde verwerking definieert de maximale grootte van de buffer een ondergrens voor de vertraging bij transmissie van een gecomprimeerde stroom.

- **complexiteit**: De complexiteit van een compressietechniek is een maat voor de lengte van de implementatie. Dit kan bijvoorbeeld het aantal lijnen zijn van een software-implementatie. Deze eigenschap is niet te verwarren met de kost die vaak een tegengesteld gedrag vertoont. Dit is ook niet te verwarren met de complexiteit van een bronobject in het kader van de complexiteitstheorie.

- **universaliteit**: De universaliteit drukt uit in welke mate eenzelfde techniek toe te passen is op verschillende beeldtypes.

- **adaptiviteit**: De adaptiviteit drukt uit in welke mate de techniek de geschatte probabiliteiten aanpast aan elk nieuw beeld.

- **schaalbaarheid**: De schaalbaarheid drukt de afhankelijkheid uit van de efficiëntie en de kost als functie van de grootte van het bronobject.

- **bewerkbaarheid**: Bij sommige technieken is het mogelijk een aantal beeldoperaties rechtstreeks uit te voeren in het getransformeerd of gecodeerd domein.

- **idempotentie**: Een verlieshebbende techniek is idempotent indien het verlies na één compressie-decompressie-iteratie gelijk is aan het verlies na meerdere identieke iteraties.

Daar waar de nadruk vroeger vooral lag op de efficiëntie en de kost van de implementatie treedt er momenteel een verschuiving op naar eigenschappen zoals universaliteit, progressiviteit en schaalbaarheid.

2.3 Basisconcepten uit de informatietheorie

2.3.1 Definities en concepten

De informatietheorie biedt een fundamenteel antwoord op de zoektocht van de datacompressie naar de ultieme compressieverhouding [42]. Het antwoord is geformuleerd in de grootheid *entropie* die zijn oorsprong vindt in de thermodynamica maar die intussen doorgedrongen is in veel takken van de wetenschap [20, 116, 221, 258].

Het discrete geheugenloze bronmodel voor een gedecorreleerd beeld beschouwt de individuele pixels als onafhankelijke uitkomsten van een toevalsveranderlijke X. De geassocieerde probabiliteitsmassafunctie is $p(x) = \Pr[X = x]$ voor alle $x \in \mathcal{X}$, waarbij $\mathcal{X}$ de verzameling is van alle waarden die de toevalsveranderlijke X kan aannemen. Een niet-adaptieve techniek maakt

vooraf een veronderstelling voor de probabiliteiten $p(x)$. Semi-adaptieve en adaptieve technieken maken een schatting $\hat{p}(x)$ die gebaseerd is op het bronobject.

De *Shannon-entropie* of kortweg *entropie* (*E.* entropy) $H(X)$ van de discrete toevalsveranderlijke X is een maat voor de hoeveelheid informatie vervat in X en is gedefinieerd als

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x),$$

of ook $H(X) = \mathrm{E}[-\log p(X)]$ waarbij $\mathrm{E}[\cdot]$ de verwachtingswaarde uitdrukt. Als grondtal van de logaritmische functie wordt 2 genomen en de entropie wordt uitgedrukt in *bits*. Er kan gemakkelijk aangetoond worden dat $0 \leq H(X) \leq \log l(\mathcal{X})$, waarbij $l(\cdot)$ het aantal elementen voorstelt in een verzameling.

De *relatieve entropie* (*E.* relative entropy) of *Kullback-Leibler-afstand* $D(p \,\|\, q)$ van een probabiliteitsmassafunctie $p(x)$ ten opzichte van een andere massafunctie $q(x)$ is gedefinieerd als

$$D(p \,\|\, q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}.$$

Het is een maat voor de afwijking in de entropie als gevolg van een afwijkende massafunctie wat zich uit in het feit dat $D(p \parallel p) = 0$. In het kader van entropiecodering is deze grootheid een maat voor de penalisatie in bitdiepte indien probabiliteiten verkeerd ingeschat worden.

Stelt Y een tweede toevalsveranderlijke voor en is $p(x, y)$ de gezamenlijke probabiliteitsmassafunctie, dan is de *gezamenlijke entropie* (*E.* joint entropy) $H(X, Y)$ van het paar discrete toevalsveranderlijken (X, Y) gedefinieerd als

$$H(X, Y) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) \log p(x, y),$$

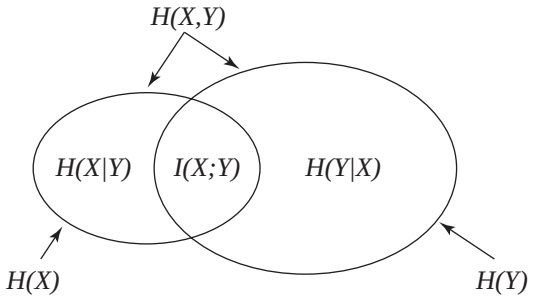
of ook $H(X, Y) = \mathrm{E}[-\log p(X, Y)]$.

Verder is de *conditionele entropie* (*E.* conditional entropy) $H(Y|X)$ gedefinieerd als

$$H(Y|X) = \sum_{x \in \mathcal{X}} p(x) H(Y|X = x),$$

wat equivalent is aan $H(Y|X) = \mathrm{E}[-\log p(Y|X)]$. Het is een maat voor de hoeveelheid informatie vervat in Y eenmaal X gekend is.

Er kan eenvoudig aangetoond worden dat $H(X, Y) = H(X) + H(Y|X)$, een eigenschap die gekend staat als de *kettingregel* (*E.* chain rule). Een onmiddellijk en belangrijk gevolg is dat "conditioneren de entropie verlaagt", dus $H(Y|X) \leq H(Y)$ met gelijkheid enkel indien X en Y onafhankelijk zijn.



Figuur 2.2: Illustratie van de relaties tussen enkelvoudige entropie, de gezamenlijke entropie, de conditionele entropie en de mutuele informatie.

Worden de marginale probabiliteitsfuncties van X en Y gegeven door respectievelijk $p(x)$ en $p(y)$ dan is de *mutuele informatie* (*E.* mutual information) gegeven door

$$I(X;Y) = \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)},$$

wat equivalent is aan de relatieve entropie $D(p(x,y) \,\|\, p(x)p(y))$ van de gezamenlijke massafunctie ten opzichte van de productmassafunctie. De mutuele informatie is een maat voor de hoeveelheid informatie die één toevalsveranderlijke bezit over de andere. Het is ook de vermindering in de onzekerheid van één toevalsveranderlijke als gevolg van kennis van de andere. De interactie tussen de respectieve grootheden wordt grafisch geïllustreerd in figuur 2.2.

Voor de modellen die gebruik maken van contexten of toestanden volstaat het niet langer om de individuele pixels te beschouwen als uitkomsten van een onafhankelijke toevalsveranderlijke. Een toevalsproces $\{X_i\}$ met tijdsindex i vormt een betere beschrijving. De *entropiesnelheid* of het *entropiedebiet* (*E.* entropy rate) kan gedefinieerd worden op twee manieren,

$$\begin{aligned} H(\mathcal{X}) &= \lim_{n\to\infty} \frac{1}{n} H(X_1, X_2, \ldots, X_n), \\ H'(\mathcal{X}) &= \lim_{n\to\infty} H(X_n | X_{n-1}, X_{n-2}, \ldots, X_1), \end{aligned}$$

waarbij kan aangetoond worden dat $H(\mathcal{X}) = H'(\mathcal{X})$ voor een stationair toevalsproces. Voor een markovproces kan deze uitdrukking herleid worden als functie van de stationaire bezetting en de transitieprobabiliteiten.

2.3.2 Limieten voor datacompressie

Deze grootheden vormen de basis van een aantal fundamentele wetmatigheden waar alle compressietechnieken aan voldoen [2]. Het belangrijkste resultaat is ongetwijfeld dat de entropie H een ondergrens vormt voor de verwachte codelengte, dus

$$\sum_{x \in \mathcal{X}} p(x) l_C(x) \geq H(X),$$

voor elke uniek decodeerbare code C die aan waarde x een codewoord met lengte $l_C(x)$ toekent. In het volgend hoofdstuk komen codes aan bod die deze ondergrens arbitrair dicht benaderen.

Statische technieken gaan uit van voorgedefinieerde codes $l_C^{(q)}$ die een benaderde massafunctie $q(x)$ vooronderstellen. Als gevolg hiervan neemt de ondergrens toe tot

$$\sum_{x \in \mathcal{X}} p(x) l_C^{(q)}(x) \geq H(X) + D(p \,\|\, q),$$

wat een interpretatie geeft aan de relatieve entropie $D(p \,\|\, q)$.

Adaptieve compressietechnieken genereren doorgaans een voortdurend wijzigende schatting $\hat{p}(x)$ van de probabiliteitsmassafunctie. Voor een gegeven sequentie $(x_1, x_2, \ldots, x_n)$ geeft dit aanleiding tot de *empirische entropie* (*E.* empirical entropy)

$$h = \frac{1}{n} \sum_{i=1}^{n} - \log \hat{p}(x_i),$$

waarbij de sommatie verloopt over de indexering en niet over de bronverzameling $\mathcal{X}$. Deze grootheid is een mathematische constructie die in relatie staat tot de compressieverhouding maar geen interpretatie heeft in de informatietheorie.

De eigenschap dat conditionering de entropie verlaagt vormt de basis van alle technieken die gebaseerd zijn op contextmodellering. Stelt X de pixelwaarde voor dan zal $H(X|Y) \leq H(X)$ en dus is het de bedoeling een gerelateerde maar causale toevalsveranderlijke Y te definiëren die een maximale daling in de entropie veroorzaakt. Al kan de entropie in theorie niet toenemen, toch moet het aantal waarden in $\mathcal{Y}$ in de praktijk beperkt worden om een effectieve afname te bekomen.

2.3.3 De typische verzameling

Het modelleren van het gehele beeld als uitkomst van een toevalsveranderlijke leidt tot een interessante invalshoek. Beschouw de toevalsrij $X_1, X_2, \ldots, X_n$ waarbij de toevalsveranderlijken X_i onafhankelijk en identiek verdeeld zijn

volgens de probabiliteitsmassafunctie $p(x)$. Stelt $p(X_1, X_2, \ldots, X_n)$ de gezamenlijke probabiliteit voor van de gehele rij, dan formuleert de *asymptotische equipartitie-eigenschap* (*E.* asymptotic equipartition property) of AEP dat $p(X_1, X_2, \ldots, X_n)$ in waarschijnlijkheid arbitrair dicht zal liggen bij 2^{-nH}, of

$$-\frac{1}{n}\log p(X_1, X_2, \ldots, X_n) \to H(X) \qquad \text{in waarschijnlijkheid},$$

waarbij $\to$ de convergentie uitdrukt.

Dit laat toe de verzameling sequenties $(x_1, x_2, \ldots, x_n) \in \mathcal{X}^n$ van lengte n voor arbitrair kleine ϵ op te delen in twee disjuncte deelverzamelingen: de *typische* verzameling $A_\epsilon^{(n)}$ en de *atypische* verzameling $\bar{A}_\epsilon^{(n)} = \mathcal{X}^n \setminus A_\epsilon^{(n)}$. De typische verzameling bevat alle sequenties waarvoor

$$2^{-n(H(X)+\epsilon)} \leq p(x_1, x_2, \ldots, x_n) \leq 2^{-n(H(X)-\epsilon)},$$

dus hun probabiliteit van optreden stemt overeen met de entropie.

Voor voldoende grote n kan eenvoudig aangetoond worden dat $\Pr[A_\epsilon^{(n)}] > 1-\epsilon$, dat de probabiliteit van een typische sequentie $p(x_1, x_2, \ldots, x_n) = 2^{-nH(X)\pm\epsilon}$ en dat $l(A_\epsilon^{(n)}) \leq 2^{n(H(X)+\epsilon)}$. Met andere woorden, de typische verzameling A_ϵ^n heeft als probabiliteit nagenoeg 1, alle elementen binnen de typische verzameling zijn even waarschijnlijk en het aantal elementen bedraagt nagenoeg 2^{nH}.

Deze eigenschap legt de basis voor een holistische aanpak voor datacompressie die intuïtief aantrekkelijk is maar in de praktijk niet uit te voeren. De verzameling $\mathcal{X}^n$ van alle sequenties van lengte n wordt opgesplitst in de typische verzameling en de atypische verzameling. De elementen in de typische verzameling worden geordend en elk krijgt een index van lengte $n(H+\epsilon)+1$ bits toegewezen. Analoog hieraan krijgt elk element in de atypische verzameling een index van lengte $n\log l(\mathcal{X})+1$ bits toegewezen. Verder krijgen de elementen een additionele '0'-bit of '1'-bit als prefix toegewezen naarmate ze behoren tot respectievelijk de typische of de atypische verzameling. Deze aanpak leidt tot een compressietechniek die met een waarschijnlijkheid van nagenoeg 1 alle sequenties comprimeert tot lengte nH.

Tegelijkertijd laat deze invalshoek toe een eigenschap te formuleren waar elk optimaal compressiealgoritme moet aan voldoen. Met waarschijnlijkheid nagenoeg 1 moet elke willekeurige maar korte bitsequentie die aangeleverd wordt aan het decodeerproces aanleiding geven tot een representatieve bronsequentie. Zelfs voor de meest geavanceerde technieken is dit momenteel niet of nauwelijks het geval.

Het aantal elementen in de typische verzameling ligt aan de basis van de logaritmische opbrengstcurve voor compressie. De efficiëntie van een techniek

heeft een lineair verband met de opslagkost of transmissietijd, maar verhoudt zich logaritmisch tot de grootte van de typische verzameling. Dit betekent dat een verdubbeling van de efficiëntie de grootte van de nieuwe typische verzameling herleidt tot de vierkantswortel van de oorspronkelijke typische verzameling. Vice versa, indien een techniek de typische verzameling halveert qua grootte, dan levert dit slechts één bit winst op qua compressielengte. Om deze reden wordt in de praktijk een efficiëntietoename van 10% vaak al als significant ervaren. Dit staat in schril contrast met de wet van Moore die van toepassing is op de groeicurve van de bronmiddelen.

2.3.4 Subjectieve entropie

Verliesloze compressietechnieken voor tekst hebben weinig of geen voorkennis van een taal en de lengte van de tekst is te klein om een zinvolle benadering te bekomen van de limiet in de definitie van het entropiedebiet. De vraag stelt zich welke het intrinsieke entropiedebiet is van een taal indien de volledige basiskennis van de taal op voorhand gekend is. De *subjectieve entropie* (*E.* subjective entropy) beantwoordt deze vraag aan de hand van een subjectieve grootheid die gebaseerd is op de hypothese dat proefpersonen een representatieve kennis hebben van een taal. Deze krijgen een tekst voorgelegd en moeten gokken welke de ontbrekende letters zijn. De resultaten van hun gokwerk worden gerangschikt en statistisch verwerkt, wat aanleiding geeft tot de subjectieve entropie.

Een dergelijk experiment leert dat de subjectieve entropie van Engelse tekst nagenoeg 1.30 bits per karakter bedraagt [41, 222]. Deze waarde vormt een benadering voor het intrinsieke entropiedebiet die een onbegrensd groot maar eindig model zou bereiken voor ongeziene teksten. Ter vergelijking, een gevorderd machinemodel dat een aanzienlijke hoeveelheid Engelse tekst als training gebruikt, haalt circa 1.73 bits per pixel voor dezelfde tekst [241].

Voor beelden en andere bronnen leidt de aanpak tot minder representatieve resultaten wegens de inherent continue aard van de voorstelling. Voor een lageresolutiebeeld met zestien grijswaarden is een visueel verliesloze compressieverhouding van 2 tot 4 haalbaar [126].

2.4 Verband met de complexiteitstheorie

Complexiteit is een begrip dat vele interpretaties kent. Als evaluatiecriterium voor compressietechnieken is het een maat voor de lengte van de implementatie. Het is niet te verwarren met de kost die een maat is voor de vereiste hoeveelheid geheugen en processortijd. Maar in het kader van de complexi-

teitstheorie heeft het een veel diepere betekenis die de notie van entropie nog voorafgaat.

2.4.1 Algoritmische complexiteit

De *algoritmische* (*E.* algorithmic) of *Kolmogorov-complexiteit* $K_U(x)$ van een object x voor een universele computer U is gedefinieerd als de lengte van het kortste computerprogramma dat bij uitvoering op de universele computer U exact x genereert in eindige tijd en vervolgens stopt, dus

$$K_U(x) = \min_{p \in \mathcal{P}} \{l(p);\ U(p) = x\},$$

waarbij $\mathcal{P}$ de verzameling voorstelt van alle programma's en $U(p)$ het resultaat van de uitvoering van het programma p op de computer U.

De definitie gaat terug op de beschrijving van het object en om die reden valt het onder de noemer *beschrijvende complexiteit* (*E.* descriptive complexity). In tegenstelling tot de entropie brengt de beschrijvende complexiteit ook de lengte van het programma dat het object reconstrueert in rekening. Het betreft hier de informatie vervat in de "conventie" van figuur 2.1. Voorgaande uitdrukking mag dan wel een eenduidige definitie zijn van de Kolmogorov-complexiteit, het betekent helaas ook dat de Kolmogorov-complexiteit een niet-berekenbare grootheid is. Het kan oneindig lang duren om na te gaan of een programma p al dan niet na een eindige tijdsduur een uitkomst zal genereren en stoppen.

De beschrijvende complexiteit heeft een vrij dichte relatie met de informatietheorie. Voor een toevalsproces $\{X_i\}$ van lengte n waarbij de toevalsveranderlijken X_i onafhankelijk en identiek verdeeld zijn geldt het verband

$$\mathrm{E}[K_U(X^n|n)/n] \rightarrow H(X),$$

waarbij $K_U(X^n|n)$ de conditionele Kolmogorov-complexiteit van de sequentie voorstelt voor een universele computer U met kennis van n. Dit betekent dat de complexiteit per toevalsveranderlijke de entropie benadert.

Net zoals bij de typische verzameling leent deze conceptuele grootheid tot een algoritme voor een optimale compressietechniek onder de randvoorwaarde van een begrensde tijdsduur T en een gegeven computer. Om een gegeven sequentie x te comprimeren, voert een computer achtereenvolgens alle programma's p uit van lengte 1, 2, ..., waarbij elk programma voortijdig wordt afgebroken indien het niet stopt binnen de begrensde tijdsduur T. Is de uitkomst van een programma p gelijk aan x, dan stopt het algoritme en is de

beschrijvende complexiteit $K(x)$ gegeven door $l(p)$. De beschrijvende complexiteit $K(x)$ voldoet aan de bovengrens

$$K(x) < K_U(x|l(x)) + \log^* l(x) + c,$$

met $\log^* \alpha = \log \alpha + \log\log \alpha + \log\log\log \alpha + \cdots$ en c een constante die afhankelijk is van de computer. Deze laatste grootheid stemt overeen met het programma "print x van lengte $l(x)$ en stop". Indien T voldoende groot gekozen wordt, dan levert de aanpak altijd een resultaat op in eindige tijd. Helaas is deze eindige tijd nog altijd onrealistisch groot voor zelfs maar de kleinste sequenties x.

Verder in dit werk komen technieken aan bod die gebaseerd zijn op Boolese minimalisatie. De modelvorming gaat daar uit van de zoektocht naar het kortste equivalent programma op een eenvoudige binaire computer, al is de aanpak heuristisch en niet exhaustief.

2.4.2 Stochastische complexiteit

De premisse van een universele computer wordt later opgegeven en een nieuwe definitie van complexiteit vindt zijn oorsprong in de statistische wereld. In referentie tot een gegeven code C met geassocieerde probabiliteitsdistributie p wordt de *stochastische complexiteit* van een geobserveerde sequentie x gedefinieerd als $-\log \Pr[x|C]$. Vervolgens wordt deze grootheid geminimaliseerd over een verzameling modellen $M \in \mathcal{M}$, waarbij elk model M een eigen waarschijnlijkheid $p(M)$ en een geassocieerde code C_M heeft. Deze aanpak heet het *principe van de minimale beschrijvingslengte* (*E.* minimum description length principle) of MDL-principe [185, 187]. De minimale beschrijvingslengte van x bestaat uit de beschrijvingslengte van x gegeven C_M en de beschrijvingslengte van M in $\mathcal{M}$.

Deze aanpak laat toe een aantal modelklassen met een variërend aantal parameters onderling te evalueren. Het MDL-principe genereert een kader dat zich uitstekend leent tot modelselectie op basis van het vergelijken van de complexiteit aan de hand van statistische waarnemingen. In tegenstelling tot de Kolmogorov-complexiteit is de minimale beschrijvingslengte wel een berekenbare grootheid. Het afbakenen van een ruimte van modelklassen blijkt hierbij een uitdaging te zijn. De fundamenten van MDL maken heden ten dage furore onder de term *universele modellering* (*E.* universal modeling) [188].

2.5 Besluit

Het algemeen schema voor verliesloze beeldcompressie kent twee grote bouwblokken: de modelvorming en de codering. De modelvorming gaat op zoek naar een werkbaar onderliggend universeel model dat zo goed mogelijk de redundantie onderschept. Tegenover dit model vertoont het waargenomen bronobject een maximale waarschijnlijkheid van optreden. Dit vormt het hart van het compressiealgoritme en de resultaten ervan leiden tot nieuwe inzichten in verwante wetenschappen zoals tekstherkenning en financiële modellering. De modelvorming genereert probabiliteitsschattingen die vervolgens door de codering gebruikt worden om de representatielengte te verkleinen en op deze manier de eigenlijke compressie te bekomen. Deze reductie is aan een aantal fundamentele grenzen onderworpen die volgen uit het model. De decompressie bestaat uit de decodering en de reconstructie.

De informatietheorie vormt de basis voor de modelvorming en legt de brug naar de codering. Ze definieert de fundamentele grootheid entropie die een maat is voor de informatie of de niet-voorspelbaarheid van de gegevens. Ze laat bovendien toe om datacompressie te benaderen vanuit een holistisch standpunt, waarbij compressietechnieken opgesteld worden op basis van de verzameling van alle typische beelden eerder dan op basis van numerieke bewerkingen op de eigenlijke pixels. Dit leidt tot een aantal inzichten en fundamentele grenzen voor de verliesloze compressie.

Verliesloze compressie gaat nauw samen met de notie van de beschrijvende of de algoritmische complexiteit. Beiden gaan op zoek naar de kortste beschrijving van een gegeven object, zij het tegenover een verschillend referentiekader. Het principe van de minimale beschrijvingslengte laat toe modelklassen met een verschillend aantal parameters objectief te vergelijken en leidt tot de notie van universele modellering.

Hoofdstuk 3

Entropiecodes

3.1 Inleiding

De entropiecodering vormt één van de wezenlijke bouwblokken in het algemeen compressieschema voor verliesloze compressie. Dit bouwblok staat in voor het genereren van een equivalente maar kortere voorstelling van de oorspronkelijke sequentie en is hiermee als enig bouwblok verantwoordelijk voor de eigenlijke reductie in de lengte van de voorstelling.

Beschouwen we een beeld als één mogelijke uitkomst van een toevalsproces, dan toont de informatietheorie aan dat er een ondergrens is voor de verwachte lengte van elke equivalente voorstelling [42]. Een ideale entropiecoder slaagt er in deze ondergrens arbitrair dicht te benaderen. Dit gebeurt door korte codewoorden toe te kennen aan veel voorkomende bronwaarden en vice versa, zonder de unieke decodeerbaarheid in gedrang te brengen. De morsecode is historisch één van de oudste codes die deze laatste eigenschap heeft.

In dit hoofdstuk tonen we aan hoe entropiecodes ontstaan zijn en welke hun belangrijkste eigenschappen zijn. Paragraaf 3.2 geeft een strikte definitie van een code evenals een aantal voorbeelden. Vervolgens beschrijft paragraaf 3.3 hoe codes geclassificeerd kunnen worden volgens een aantal uiteenlopende eigenschappen. Eindige entropiecodes zijn voor ons de belangrijkste en ze worden dan ook uitvoerig behandeld in paragraaf 3.4. De klassieke huffman- en aritmetische codes, een aantal varianten en hun implementaties komen uitgebreid aan bod. Paragraaf 3.5 beschrijft een aantal codes die gedefinieerd zijn voor onbegrensde bronwaarden, zoals de golomb- en ricecodes. Tot slot vat paragraaf 3.6 de belangrijkste codes nog eens samen. De aanpak in dit hoofdstuk start vrij rigoreus en behandelt initieel enkel tijdsonafhankelijke codes. Gaandeweg verschuift de aandacht naar codes die voor de praktijk geschikt zijn en neemt het belang van de theoretische onderbouw af.

3.2 Definitie van een code

3.2.1 Definitie en nomenclatuur

Enerzijds stelt X een discrete toevalsveranderlijke voor die waarden kan aannemen in $\mathcal{X}$. Anderzijds is A een alfabet dat bestaat uit een eindig aantal symbolen. Elke opeenvolging van een eindig aantal symbolen is een *woord* en A^* stelt de verzameling voor van alle *woorden*. Codering heeft tot doel aan elke waarde die X kan aannemen een voorstelling met symbolen uit A te koppelen. Bij wijze van voorbeeld kan X een karakter uit een tekst voorstellen en kan A het binaire alfabet $\{0, 1\}$ zijn. Dan is $\mathcal{X} = \{a, b, \ldots, z\}$ en is $A^* = \{\Lambda, 0, 1, 00, 01, 10, \ldots\}$, waarbij Λ het woord van lengte 0 voorstelt.

Een *broncode* C (*E.* source code) wordt gedefinieerd als een afbeelding van $\mathcal{X}$ naar A^*. De waarden van $\mathcal{X}$ worden de *bronwaarden* genoemd en de symbolen van A de *codeletters*. Het *codewoord* behorend bij de bronwaarde $x \in \mathcal{X}$ wordt voorgesteld door $C(x)$ met als lengte $l_C(x)$. De verzameling van alle codewoorden wordt voorgesteld door $A_C^* \subset A^*$. Het *codeboek* beschrijft de samenhang tussen de bronwaarden en de codewoorden. De bronwaardenverzameling $\mathcal{X}$ wordt ook soms het *bronalfabet* genoemd en het alfabet A het *codealfabet*. Deze definitie betreft enkel *statische* of *tijdsonafhankelijke* codes. Verder in dit hoofdstuk wordt ze uitgebreid naar codes waarvan de afbeelding van $\mathcal{X}$ naar A^* expliciet afhangt van de tijd.

Het *coderen* van een *sequentie*, dit is een rij bronwaarden uit $\mathcal{X}$, vervangt iedere bronwaarde x door het corresponderende codewoord $C(x)$. Het resultaat wordt het *bericht* genoemd. Deze terminologie verwijst naar de verspreiding van de informatie. De omgekeerde stap, het *decoderen*, zet een bericht om in een sequentie bronwaarden op zo'n manier dat het coderen van de resulterende sequentie opnieuw het oorspronkelijke bericht oplevert. Decoderen is niet altijd mogelijk en als het mogelijk is, levert het niet altijd een uniek resultaat.

3.2.2 Voorbeelden

Een voor de hand liggend voorbeeld van een code is de *ASCII-code* die veel gebruikt wordt in de computerwereld [14, 209]. Hier bestaat $\mathcal{X}$ uit de 128 meest voorkomende controle- en tekstkarakters en is $A = \{0, 1\}$. Zo wordt '*a*' voorgesteld door het codewoord '0100 0001.' Ieder codewoord is verschillend en bevat 8 bits waardoor slechts 128 van de 256 mogelijke codewoorden gebruikt worden. De eerste bit fungeert evenwel als pariteitsbit, maar deze functionaliteit wordt niet overal op dezelfde manier geïmplementeerd. In de praktijk wordt deze niet gebruikt: hij wordt op '0' ingesteld en de 128 andere

codewoorden worden toegekend aan andere waarden. De 7-bits ASCII-code bestaat dan ook uit de klassieke ASCII-codewoorden zonder de eerste bit. Het decoderen van deze code is eenvoudig omdat ieder codewoord even lang is.

De toekenning van telefoonnummers aan abonnees is een minder voor de hand liggende code. Voor de eenvoud nemen we aan dat iedere abonnee over juist één telefoonnummer beschikt. De abonnees vormen de bronverzameling $\mathcal{X}$ en iedereen kan elke andere abonnee bereiken door een eindig aantal toetsen uit $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ op een gepaste manier te combineren. De lengte van de codewoorden is hier niet langer constant maar toch is de code zo opgebouwd dat ieder codewoord uniek en onmiddellijk decodeerbaar is: bij het indrukken van een nummer moet geen speciaal 'einde-codewoord'-symbool of 'enter'-toets gedrukt worden om duidelijk te maken dat het nummer volledig gevormd is.

De *morsecode* is een code waarvan de functionaliteit al dichter aanleunt bij de doelstelling van compressie: ze is uniek decodeerbaar en past de lengte van de bronwaarden aan aan hun probabiliteit van optreden. Het alfabet bestaat uit de symbolen '$\bullet$' en '$-$' en de codewoorden worden gescheiden door een spatie, meestal voorgesteld door '$\sqcup$.' De codewoorden verschillen van lengte en de code is zo geconstrueerd dat de meest voorkomende letters afgebeeld worden op de kortste codewoorden. Op deze manier wordt de verwachte lengte van de te coderen tekstsequentie kleiner: de tekstsequentie wordt gecomprimeerd.

3.3 Classificatie

Alvorens de codes voor compressiedoeleinden te bespreken, volgt hier een overzicht van de verschillende criteria die kunnen gehanteerd worden om codes te classificeren.

3.3.1 Classificatie volgens decodeerbaarheid

Niet alle codes die voldoen aan bovenstaande definitie zijn nuttig. Meestal is unieke decodeerbaarheid gewenst en hiervoor moet de code voldoen aan een aantal voorwaarden.

Een code is *niet-singulier* of *regulier* (*E.* non-singular) indien verschillende elementen uit $\mathcal{X}$ afgebeeld worden op verschillende codewoorden, dus als $x \neq y$, moet ook $C(x) \neq C(y)$.

Met iedere waarde van $\mathcal{X}$ wordt dus een uniek codewoord geassocieerd en deze eigenschap laat toe dat ieder codewoord op een unieke manier kan gedecodeerd worden. Maar meestal wordt niet één enkele waarde maar een sequentie van waarden gecodeerd. Om deze sequentie uniek te kunnen decoderen,

zijn er nu twee keuzes mogelijk. Ten eerste kan men een niet-singuliere code gebruiken en de opeenvolgende codewoorden scheiden door speciale *scheidingstekens*, zoals bijvoorbeeld het spatiesymbool '␣' bij de morsecode. Met het oog op compressie betekent dit echter een inefficiënt gebruik van dit speciaal scheidingssymbool.

De tweede mogelijkheid rijgt de individuele woorden aaneen zonder gebruik te maken van een scheidingsteken maar dan moeten de codes voldoen aan een extra voorwaarde. Daartoe wordt eerst de *uitbreiding* C^* (*E.* extension) van een code C gedefinieerd als de afbeelding van $\mathcal{X}^*$, de verzameling van alle eindige sequenties van waarden uit $\mathcal{X}$, naar A^*, de verzameling van alle eindige woorden opgebouwd uit symbolen van A, volgens

$$C(x_1 x_2 \cdots x_n) = C(x_1)C(x_2)\cdots C(x_n), \tag{3.1}$$

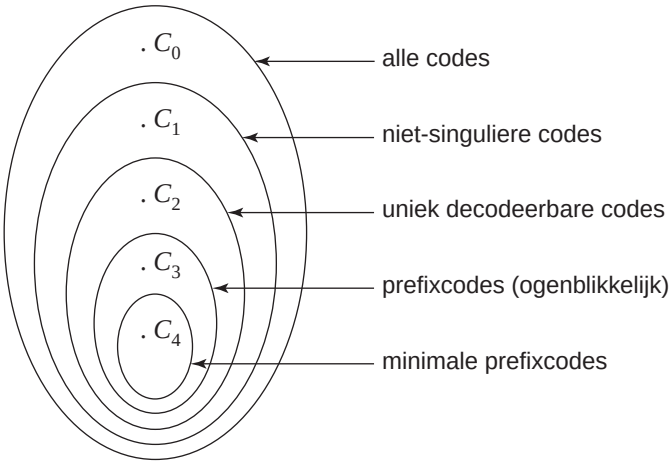
waarbij $C(x_1)C(x_2)\cdots C(x_n)$ de *aaneenrijging* (*E.* concatenation) is van de respectieve codewoorden. Een code wordt *uniek decodeerbaar* (*E.* uniquely decodable) genoemd indien haar uitbreiding niet-singulier is.

Een uniek decodeerbare code zet dus elke mogelijke eindige sequentie om in een verschillende gecodeerde sequentie. Helaas zijn combinaties van codes en sequenties mogelijk waar heel het bericht in beschouwing moet genomen worden om de eerste bronwaarde te reconstrueren. Dit probleem doet zich niet voor bij *ogenblikkelijke* (*E.* instantaneous) codes of *prefixcodes*. Een *prefix* of *voorvoegsel* van een codewoord '$a_1 a_2 \cdots a_k$' is elk woord van de vorm '$a_1 a_2 \cdots a_j$' met $j \leq k$. Voor een *eigenlijke* prefix moet bovendien $j \neq k$. Bij een prefixcode is geen enkel codewoord een prefix (een voorvoegsel) van een ander codewoord. Dankzij deze bijkomende eigenschap kunnen de opeenvolgende codewoorden binnen een bericht ogenblikkelijk herkend en gedecodeerd worden.

Een *minimale* prefixcode is een prefixcode die voldoet aan de volgende eigenschap: indien a een eigenlijke prefix is van een codewoord, dan is ab ofwel een codewoord, ofwel een eigenlijke prefix van een codewoord, en dit voor elke $b \in A$. Elke prefixcode kan geminimaliseerd worden door van elk codewoord precies die laatste bits te verwijderen die de prefixeigenschap niet in het gedrang brengen.

De hierboven beschreven eigenschappen van codes zijn grafisch te interpreteren door de codewoorden te situeren in een *codeboom*. Dit is een begrensde binaire boom waarbij bronwaarden geassocieerd worden met bepaalde knooppunten.

Figuur 3.1 illustreert hoe deze klassen elkaar omvatten. Een voorbeeld van een code uit elke klasse voor $\mathcal{X} = \{a, b, c, d\}$ en $A = \{0, 1\}$ wordt er gesitueerd en hun respectieve codewoorden zijn weergegeven in tabel 3.1.

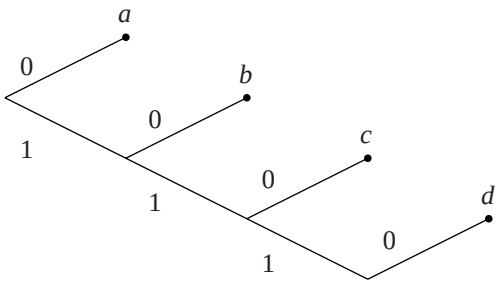


Figuur 3.1: Classificatie van de verschillende types codes met situering van de voorbeelden C_0, C_1, C_2, C_3 en C_4 uit tabel 3.1.

Code C_0 is duidelijk singulier want alle waarden van $\mathcal{X}$ worden op hetzelfde codewoord '0' afgebeeld. Code C_1 is regulier, maar niet uniek decodeerbaar omdat de codesequentie '010' zowel als '*b*' als '*ca*' kan gedecodeerd worden. Code C_2 is uniek decodeerbaar, maar omdat '0' een prefix is van '01' is ze niet ogenblikkelijk. Codes C_3 en C_4 zijn beide ogenblikkelijke codes omdat geen enkel codewoord een prefix is van een ander. In tegenstelling tot code C_3 heeft code C_4 de bijkomende eigenschap dat ze minimaal is. Immers, voor code C_3 is '111' een eigenlijke prefix van '1110', maar is '1111' noch een codewoord noch een eigenlijke prefix van een codewoord. De codeboom horend bij code C_3 wordt voorgesteld in figuur 3.2.

Tabel 3.1: Voorbeeld van codes uit de verschillende klassen van decodeerbaarheid (afgekort als dec.) met $\mathcal{X} = \{a, b, c, d\}$ en $A = \{0, 1\}$.

$\mathcal{X}$	C_0 singulier	C_1 regulier niet uniek dec.	C_2 uniek dec. geen prefix	C_3 prefix niet minimaal	C_4 minimaal
a	0	0	0	0	0
b	0	010	01	10	10
c	0	01	011	110	110
d	0	10	0111	1110	111



Figuur 3.2: De codeboom voor code C_3. De codewoorden zijn gemarkeerd met '•' en de corresponderende bronwaarde.

De unieke decodeerbaarheid van code C_2 is hier nog eenvoudig na te gaan, maar voor grotere codes is dit minder evident. Voor die gevallen bestaat er een eindige test die verzamelingen construeert van alle mogelijke suffixen en via systematische eliminatie de unieke decodeerbaarheid nagaat [212].

3.3.2 Classificatie volgens toepassing

Door de toenemende integratie van computers is er in de loop van de voorbije decennia een grote nood ontstaan aan efficiënte codes voor verschillende doeleinden. Nagenoeg elke machinevoorstelling van informatie gebruikt $A = \{0, 1\}$ en de resulterende codes worden *binaire* codes genoemd.

Elementaire code

Een *elementaire* code kent aan elke waarde een uniek codewoord van gelijke, maar minimale, lengte toe. Indien $\mathcal{X}$ bestaat uit l verschillende waarden en A uit b verschillende symbolen, dan bestaat elk codewoord uit precies $\lceil \log_b l \rceil$ codeletters. Indien $l = b^q$ met q een natuurlijk getal, is de code *volledig*. Indien $b = 2$ spreekt men van een *binaire* elementaire code. Coderen en decoderen zijn triviale opdrachten en gebeuren aan de hand van een tabel. Het doel van deze codes is veelal een eindige verzameling waarden voorstellen op een eenvoudige en gestandaardiseerde manier. De bovenvermelde 7-bits ASCII-code vormt hier een goed voorbeeld van. Een ander voorbeeld wordt gegeven door de *graycodering*: de opeenvolgende codewoorden toegekend aan een geordende verzameling bronwaarden zijn allemaal even lang en verschillen slechts in één bitpositie [84].

De benaming "*de* binaire code" wordt gebruikt voor een binaire elementaire code van voldoende lengte waarvan de leidende '0'-bits weggelaten wor-

den, dus $\{1, 10, 11, 100, 101, 110, \ldots\}$. Deze code is vergelijkbaar met de klassieke decimale voorstelling van getallen. Het is geen elementaire code meer en zelfs geen prefixcode. Ze wordt gebruikt met scheidingstekens of als bouwsteen om andere codes samen te stellen.

Een *geleidelijke* (*E.* phased in) elementaire binaire code voor l waarden onderscheidt zich van de elementaire binaire code doordat ze een aantal codewoorden inkort indien $q = \log_2 l$ geen natuurlijk getal is. Er zijn $2^{\lceil q \rceil} - l$ codewoorden van $\lfloor q \rfloor$ bits en $2l - 2^{\lceil q \rceil}$ codewoorden van $\lceil q \rceil$ bits. Welke waarden een kort codewoord en welke waarden een lang codewoord toegewezen krijgen, hangt af van de toepassing. De situatie voor $b > 2$ is iets ingewikkelder omdat het aantal codewoorden van een geleidelijke elementaire code van de vorm $k(b-1)+1$ met k een natuurlijk getal is, zodat de situatie verschilt naargelang l van die vorm is of niet.

De eenvoud van elementaire codes heeft een aantal nadelen. Ten eerste kunnen ze enkel een eindig aantal codewoorden bevatten. Ten tweede is deze voorstelling meestal niet optimaal wat de lengte van het gecodeerd bericht betreft. Ten derde is een elementaire code gevoelig voor fouten. Indien een symbool van het gecodeerd bericht door externe factoren veranderd wordt, kan dit ongedetecteerd blijven met een foutieve decodering als gevolg. Ten vierde kan iedereen die het bericht onderschept en de (eenvoudige) code kent, het ook probleemloos decoderen en de oorspronkelijke sequentie reconstrueren.

Onbegrensde codes

Het eerste probleem kan opgelost worden door gebruik te maken van *onbegrensde* codes. Deze gaan uit van een onbegrensde bronverzameling die doorgaans afgebeeld wordt op de verzameling van de natuurlijke getallen. Codes voor natuurlijke getallen komen verder in dit hoofdstuk aan bod.

Entropiecode

Het tweede probleem wordt aangepakt door gebruik te maken van *entropiecodes*. In de meeste gevallen is het, rekening houdend met de waarschijnlijkheid van optreden van de bronwaarden van $\mathcal{X}$ in een sequentie, mogelijk een code te ontwerpen die voor alle typische sequenties een korter gecodeerd bericht garandeert dan bij elementaire codes. Op deze manier wordt de redundantie, die in de meeste sequenties aanwezig is, verwijderd uit de oorspronkelijke sequentie. Deze codes kennen codewoorden van verschillende lengte toe en worden daarom ook wel *variabele-lengte-codes* genoemd. Ze vormen één van de pijlers van zowel verliesloze als verlieshebbende compressie en worden verder in dit hoofdstuk meer gedetailleerd beschreven.

Foutdetecterende en foutverbeterende codes

Om het derde probleem te vermijden, werden er *foutdetecterende* en *foutverbeterende* codes ontworpen [15,92,94,139]. In tegenstelling tot entropiecodes, voegen ze redundantie toe aan de voorstelling van de verschillende waarden. Foutdetecterende codes zijn in staat om de meest waarschijnlijke fouten te detecteren, maar niet te verbeteren. Foutverbeterende codes slagen er daarenboven in om een deelverzameling van deze fouten te verbeteren. Een elegant voorbeeld van dergelijke codes zijn *hammingcodes* [86]. Cd-spelers maken gebruik van gekruiste *Reed-Solomon-codes*, wat toelaat om tot 4000 opeenvolgende bitfouten te verbeteren [179]. Het onderzoeksdomein van de codeertheorie is ruimer en eleganter dan dat van de entropiecodes. Het is sterk verweven met het domein van eindige velden en eindige meetkunde.

Cryptografische code

Het vierde probleem tenslotte wordt aangepakt door de *cryptografische* codes. Door gebruik te maken van een *sleutel* coderen cryptografische codes de oorspronkelijke informatiestroom op zo'n manier dat het oorspronkelijk bericht bijna onmogelijk kan gedecodeerd worden indien de sleutel niet gekend is [156, 218]. Cryptografisch coderen wordt ook wel *vercijferen* genoemd. Doorgaans zijn het codes die gedefinieerd zijn voor $\mathcal{X}^*$, de uitbreiding van $\mathcal{X}$. Ze zijn immers niet gedefinieerd voor de individuele waarden die X kan aannemen, maar werken in op blokken of stromen van die waarden. Ze verwijderen geen redundantie, maar voegen er ook geen toe. Het onderzoeksdomein van de cryptografische codes is jonger en is voor een groot stuk gebaseerd op de getaltheorie. Het is opmerkelijk dat Shannon, grondlegger van de informatietheorie, en Alan Turing, stichter van de computerwetenschappen, voor een groot stuk hun inspiratie haalden uit het werk dat ze in het domein van de cryptografie verrichtten voor en tijdens de tweede wereldoorlog.

Voorbeelden

Een overzicht van deze verschillende toepassingen met een voorbeeld van een code wordt weergegeven in tabel 3.2. Elk van deze codes is ogenblikkelijk. Code C_5 is een volledige elementaire code: dit is de eenvoudigste en meest voor de hand liggende binaire voorstelling. Onder gepaste voorwaarden zal code C_6 een korter gecodeerd bericht genereren dan C_5. Het codewoord voor a is korter, maar die voor c en d zijn langer. Code C_7 is in staat een fout van één bit te detecteren, maar niet om ze te verbeteren. De derde bit kan geïnterpreteerd worden als een pariteitsbit. Dit is de eenvoudigste vorm van

Tabel 3.2: Voorbeeld van codes voor verschillende toepassingen met $\mathcal{X} = \{a, b, c, d\}$ en $A = \{0, 1\}$ of $A = \mathcal{X}$.

$\mathcal{X}$	C_5 elementair	C_6 variabele lengte	C_7 fout-detecterend	C_8 fout-verbeterend	C_9 cryptografisch
a	00	0	001	01011	*b*
b	01	10	010	10010	*c*
c	10	110	100	01100	*d*
d	11	111	111	10101	*a*

foutdetecterende codes. Alle codewoorden zijn langer dan die van de elementaire code C_5. Code C_8 stelt een hammingcode voor die fouten van één en twee bits kan detecteren, en bovendien fouten van één bit kan verbeteren. Code C_9 tenslotte is de *caesarcode*, een eenvoudige cryptografische code waarbij de elementen van $A = \mathcal{X}$ gepermuteerd worden. Efficiënte cryptografische codes zijn tijdsafhankelijk en kunnen niet als een eenvoudige relatie tussen bronwaarden en codewoorden omschreven worden.

Gecombineerde codes

Een inherent nadeel van entropiecodes is dat ze nog gevoeliger zijn voor fouten dan elementaire codes. Immers, idealiter is alle redundantie uit de code verdwenen. Dit betekent dat zodra er één enkele transmissiefout opgetreden is, het onmogelijk is om het oorspronkelijke bericht te reconstrueren [268]. Sommige codes zijn zo ontworpen dat de impact van een transmissiefout beperkt is in de tijd. Enige tijd nadat de fout opgetreden is, worden opnieuw de correcte bronwaarden gereconstrueerd. Dergelijke codes worden *zelfsynchroniserend* genoemd.

Het is mogelijk om redundantie toe te voegen aan huffmancodes zodat ze bepaalde types fouten kunnen detecteren [167]. Maar fundamenteel worden geen betere resultaten verkregen dan wanneer entropiecodering en foutcodering achtereenvolgens toegepast worden. De volgorde waarin vermelde types codering gecombineerd worden, speelt een belangrijke rol. Best wordt eerst entropiecodering toegepast, omdat dit de redundantie verwijdert. Dan volgt encryptie en tot slot foutcodering omdat er hierna geen wijzigingen meer mogen aangebracht worden.

Er is een fundamenteel onderscheid tussen de aanpak van entropiecodes enerzijds en foutdetecterende en foutverbeterende codes anderzijds. De eerste veronderstellen een niet-uniforme verdeling van de bronwaarden en maken ex-

pliciet gebruik van deze verdeling, terwijl de andere geen veronderstellingen maken omtrent hun verdeling.

3.3.3 Classificatie volgens lengte

Volgens bovenstaande definitie van een broncode wordt elke bronwaarde van $\mathcal{X}$ afgebeeld op een codewoord. Een *broncode in ruime zin* is een afbeelding van $\mathcal{X}_C^*$, een deelverzameling van $\mathcal{X}^*$, naar A^*. Een broncode in ruime zin splitst de sequentie op in subsequenties en beeldt elke subsequentie af op een codewoord overeenkomstig het codeboek. Een broncode is een broncode in ruime zin waarvan $\mathcal{X}_C^*$ beperkt is tot $\mathcal{X}$, de toegelaten subsequenties beperken zich tot individuele bronwaarden. Veelal wordt de term code gebruikt voor zowel een broncode als een broncode in ruime zin.

De lengten van de subsequenties kunnen onderling verschillen. Dit is eveneens zo voor de lengten van de codewoorden onderling. Afhankelijk van hun respectieve lengten worden vier types onderscheiden: blok-blok-codes, blok-variabel-codes, variabel-blok-codes en variabel-variabel-codes, respectievelijk afgekort als BB-, BV-, VB- en VV-codes. De term "blok" wordt gebruikt indien alle subsequenties dan wel codewoorden dezelfde lengte hebben. In het andere geval wordt de term "variabel" gebruikt.

Voor VB- en VV-codes zijn er bijkomende voorwaarden nodig om te garanderen dat elke sequentie van $\mathcal{X}^*$ gecodeerd kan worden. Een code in ruime zin is *eenduidig* (*E.* proper) indien geen enkele sequentie van $\mathcal{X}_C^*$ een prefix is van een andere sequentie van $\mathcal{X}_C^*$ en *compleet* (*E.* complete) indien elke oneindige sequentie in $\mathcal{X}$ een prefix heeft binnen $\mathcal{X}_C^*$. Voor compressie worden vooral BV-codes gebruikt, al zijn VB-codes minstens even efficiënt [287]. Een mogelijk voordeel van VB-codes is dat het decoderen heel snel kan verlopen indien de implementatie gebruik maakt van een opzoektabel. Soms worden ook VV-codes gebruikt, deze zijn meestal een combinatie van een VB- en een BV-code.

Codes C_0, C_5, C_7 en C_8 uit voorgaande tabellen zijn BB-codes. Ook de ASCII-code is een BB-code. Vanuit het standpunt van compressie zijn deze nutteloos. De andere voorbeelden zijn BV-codes. Code C_{10} en C_{11} in tabel 3.3 zijn voorbeelden van respectievelijk een VB-code en een VV-code. Beide codes kunnen elke sequentie coderen, maar indien geen verdere conventies gerespecteerd worden, is het resulterende bericht voor veel sequenties niet uniek gedefinieerd. Het decoderen is wel altijd uniek gedefinieerd. Lempel-Ziv-codes, onder andere gebruikt in het UNIX-compressieprogramma "compress," vormen een praktisch voorbeeld van VB-codes [288]. Het programma "gzip" maakt gebruik van VV-codes [209].

Tabel 3.3: Voorbeeld van broncodes in ruime zin voor $\mathcal{X} = \{a, b\}$ en $A = \{0, 1\}$.

$\mathcal{X}^*$	C_{10} variabel-blok (VB)	C_{11} variabel-variabel (VV)
aaa	00	0
bb	01	10
a	10	110
b	11	111

3.3.4 Classificatie volgens tijdsafhankelijkheid

Volgens bovenstaande definitie van een broncode wordt dezelfde bronwaarde altijd afgebeeld op hetzelfde codewoord. Dergelijke codes worden *statisch* of *tijdsonafhankelijk* genoemd.

Er worden evenwel ook codes gebruikt die hier niet aan voldoen. Een *tijdsafhankelijke* broncode $\mathcal{C} = C_1 C_2 \cdots$ beeldt elke rij bronwaarden $x_1 x_2 \cdots$ af op een bericht $\mathcal{C}(x_1 x_2 \cdots)$. Hierbij wordt elke bronwaarde x_i afgebeeld op $C_i(x_i)$, zodat het resulterende bericht $C_1(x_1)C_2(x_2)\cdots$ ontstaat. Deze codes worden ook wel *dynamisch* of *adaptief* genoemd. Hierna onderscheiden we het symbool i voor de indexering binnen een sequentie s, dus $1 \leq i \leq n$ met $n = l(s)$, en het symbool α voor de indexering over de bronwaardeverzameling $\mathcal{X}$, dus $1 \leq \alpha \leq l$ met $l = l(\mathcal{X})$.

Een tijdsafhankelijke code is dus een rij statische codes. Statische en dynamische codes zijn dan ook verschillende wiskundige entiteiten: een statische code is een afbeelding van $\mathcal{X}$ naar A^*, terwijl een dynamische code een afbeelding is van $\mathcal{X}^\infty$ naar $(A^*)^\infty$. Een dynamische code waarvan $C_i = C_0$ voor alle i is bijgevolg geen statische code, maar dit semantisch onderscheid wordt meestal genegeerd.

3.4 Entropiecodes

Zoals eerder beschreven in het algemeen compressieschema is het ontwerp van algoritmen voor verliesloze compressie opgedeeld in twee bouwblokken. Enerzijds wordt er voor een gegeven sequentie van waarden een bron geschat waarvoor deze sequentie typisch is. Anderzijds moet voor deze geschatte bron een code geconstrueerd worden die de verwachte lengte van het gecodeerd bericht minimaliseert, onder de randvoorwaarde dat het gecodeerd bericht uniek decodeerbaar of zelfs ogenblikkelijk moet zijn.

3.4.1 Kraft-ongelijkheid

Het is duidelijk dat het niet mogelijk is om aan elke bronwaarde zomaar een kort codewoord toe te kennen zonder de decodeerbaarheid te compromitteren. Code C_0 uit tabel 3.1 zal duidelijk de kortste berichten genereren, maar is zeker niet uniek decodeerbaar.

In het geval van prefixcodes staat deze eigenschap gekend als de *Kraft-ongelijkheid*. Voor iedere ogenblikkelijke code (prefixcode) met alfabet A met grootte $b = l(A)$, voldoen de lengten $k_1, k_2, \ldots, k_l$ van de codewoorden aan de ongelijkheid

$$\sum_{\alpha=1}^{l} b^{-k_\alpha} \leq 1. \tag{3.2}$$

Omgekeerd, gegeven een aantal getallen $k_1, k_2, \ldots, k_l$ die voldoen aan deze ongelijkheid, is het mogelijk een ogenblikkelijke code te construeren met deze getallen als codewoordlengten. De *uitgebreide Kraft-ongelijkheid* stelt dat deze ongelijkheid blijft gelden voor ogenblikkelijke codes met aftelbaar oneindig veel codewoorden. Een bewijs voor beide eigenschappen kan gevonden worden in [42, 129]. Uit de Kraft-ongelijkheid volgt onmiddellijk dat een code die evenveel codewoorden bevat als een volledige elementaire code maar die minstens één korter codewoord bevat, automatisch ook langere codewoorden moet bevatten.

Niettegenstaande de klasse van uniek decodeerbare codes groter is dan die van de ogenblikkelijke codes, blijft deze eigenschap gelden voor uniek decodeerbare codes [124, 151]. In dit geval spreekt men van de *McMillan-ongelijkheid*. Omdat deze ongelijkheid aan de grondslag ligt van alle grenzen qua codeervermogen, bieden uniek decodeerbare codes op het vlak van compressie geen toegevoegde waarde ten opzichte van ogenblikkelijke codes. Beide ongelijkheden samen worden dikwijls beschreven als de *Kraft-McMillan-ongelijkheid*.

Het is eenvoudig na te gaan dat codes C_2, C_3 en C_4 voldoen aan de Kraft-ongelijkheid. Enkel voor code C_4, de minimale prefixcode, gaat ook de gelijkheid op.

3.4.2 Optimale codes

Stel dat C een prefixcode is voor een gegeven discrete toevalsveranderlijke X met waarschijnlijkheidsmassafunctie $p(x) = \Pr[X = x]$ en stel dat de lengten van de codewoorden gegeven worden door $l_C(x)$, verkort genoteerd als $l(x)$. Een code C is *optimaal* of *compact* indien de verwachte lengte $L = \mathrm{E}[l] = \sum_x p(x)l(x)$ kleiner is dan of gelijk is aan de verwachte lengte

van elke andere prefixcode. Merk op dat er een wezenlijk verschil is tussen het minimaal zijn van een code (een eigenschap van de codewoorden van het codeboek), en het optimaal zijn van een code (een eigenschap die betrekking heeft op de waarschijnlijkheid van optreden van de codewoorden). Een optimale code is altijd minimaal, maar niet iedere minimale code is optimaal. Een code voor oneindige bronverzamelingen is *asymptotisch optimaal* indien L/H naar 1 convergeert als H naar oneindig gaat.

De zoektocht naar de lengten van de optimale codewoorden is een klassiek optimalisatieprobleem: minimaliseer $L = \sum_x p(x)l(x)$ over alle natuurlijke getallen $l_1, l_2, \ldots, l_n$ onder de randvoorwaarde dat $\sum_x b^{-l(x)} \leq 1$. De eis van natuurlijke getallen verwaarlozend, biedt de methode van de lagrangiaanse vermenigvuldigers hiervoor een oplossing. Enig rekenwerk levert $l^*(x) = -\log_b p(x)$ op als optimale lengten. Deze oplossing, die meestal geen natuurlijke getallen als lengte geeft, genereert een verwachte lengte

$$L^* = \sum_x p(x)l^*(x) = -\sum_x p(x)\log_b p(x) = H_b(X), \qquad (3.3)$$

met $H_b(X)$ de entropie van X uitgedrukt in basis b.

De eis dat de lengten natuurlijke getallen zijn, zorgt ervoor dat de entropie $H_b(X)$ niet altijd kan bereikt worden als minimale verwachte lengte. Daarom is de verwachte lengte $L \geq H_b(X)$, met gelijkheid als en slechts als $b^{-l(x)} = p(x)$. Dergelijke massafuncties, waarvoor $p(x) = b^{-l(x)}$ met $l(x)$ een natuurlijk getal, worden *b-adisch* genoemd. Veelal wordt $b = 2$ genomen en wordt de term *dyadisch* gebruikt.

De *redundantie* of *overtolligheid* van een code wordt gedefinieerd als $L - H$. Het is een maat voor het verschil in benodigde representatielengte en informatie. Voor een optimale code is de redundantie minimaal.

Voor de codes die hierna vermeld worden, wordt het alfabet A beperkt tot $\{0, 1\}$ en is $b = 2$. Tenzij anders vermeld, wordt de logaritme genomen met basis 2 en de entropie wordt uitgedrukt in bits. Dergelijke codes worden *binaire* codes genoemd. Ze zijn uitbreidbaar naar alfabetten waarvoor $b > 2$.

3.4.3 Shannon-Fano-codes

Onafhankelijk van elkaar stelden Shannon en Fano volgende eenvoudige procedure voor om een code te construeren [74, 221, 223].

- Sorteer alle bronwaarden x volgens afnemende probabiliteit $p(x)$.
- Deel de lijst in twee opeenvolgende deellijsten van (ongeveer) even grote waarschijnlijkheid.

X	$p(x)$					$C(x)$	$l_C(x)$
e	0.40	0	0			00	2
a	0.15		1			01	2
b	0.15	1	0			10	2
d	0.15		1	0		110	3
f	0.10			1	0	1110	4
c	0.05				1	1111	4

Figuur 3.3: Voorbeeld van de constructie van een eenvoudige Shannon-Fano-code.

- Elk codewoord uit de eerste deellijst krijgt een ‘0’-bit toegekend, elk codewoord uit de tweede deellijst krijgt een ‘1’-bit toegekend.

- Herhaal deze procedure recursief voor elke deellijst die nog meer dan één bronwaarde bevat.

Bij constructie levert dit een minimale prefixcode op die een *Shannon-Fano-code* genoemd wordt.

De constructie van een code volgens bovenstaande procedure is niet uniek, want soms zijn er twee equivalente mogelijkheden om twee deellijsten te maken van even grote waarschijnlijkheid. De verwachte lengten L van de verschillende codes die volgens deze procedure kunnen verkregen worden, kunnen afwijken van elkaar. Deze codes zijn niet altijd optimaal, maar het verschil is meestal klein. Er kan aangetoond worden dat voor deze codes $H \leq L < H + 1$, zodat er niet alleen een ondergrens, maar ook een bovengrens is. Voor dyadische en enkel voor dyadische massafuncties wordt de ondergrens $L = H$ bereikt.

Bij wijze van voorbeeld schetst figuur 3.3 de procedure voor $\mathcal{X} = \{a, b, c, d, e, f\}$ met respectieve probabiliteiten $\{0.15, 0.15, 0.05, 0.15, 0.40, 0.10\}$. De entropie van de code bedraagt $H = 2.31$ bps en de verwachte lengte $L = 2.45$ bps; de redundantie bedraagt 0.14 bps.

3.4.4 Huffmancodes

Definitie

Er kan aangetoond worden dat er een optimale code bestaat die voldoet aan de volgende drie eigenschappen: (1) als $p(x) > p(y)$, dan is $l(x) \leq l(y)$, (2) de langste twee codewoorden hebben dezelfde lengte, en (3) de langste twee codewoorden verschillen enkel in de laatste bit en corresponderen met de twee minst waarschijnlijke waarden. Niet iedere code die hieraan voldoet is optimaal, zie bijvoorbeeld de Shannon-Fano-code uit figuur 3.3.

Voortbouwend op deze eigenschappen stelde Huffman een recursief algoritme op dat een optimale code genereert [105].

- Sorteer alle bronwaarden x volgens afnemende probabiliteit $p(x)$.
- Zoek de twee minst waarschijnlijke waarden uit de lijst en ken een '0'-bit toe aan de ene waarde en een '1'-bit aan de andere.
- Vervang beide waarden door een nieuwe waarde met als probabiliteit de som van de individuele probabiliteiten.
- Herhaal deze procedure totdat er slechts één enkele waarde overblijft.

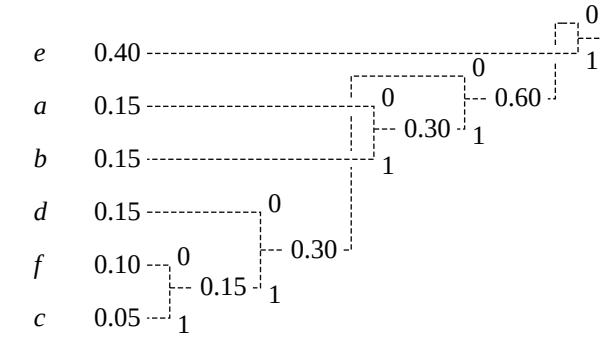
Op deze manier wordt een *huffmanboom* verkregen en de codewoorden kunnen eenvoudig afgeleid worden door de boom te doorlopen vanaf de wortel tot de bladeren, dit zijn de bronwaarden, en hierbij de individuele bits samen te voegen. De resulterende code is bij constructie een minimale prefixcode. Een dergelijke code, maar ook iedere code die dezelfde lengten heeft voor zijn codewoorden, wordt een *huffmancode* genoemd.

Eenmaal het codeboek opgesteld is, verloopt het coderen bijzonder snel omdat geen rekenkundige bewerkingen meer moeten uitgevoerd worden. Het decoderen gebeurt door de boom vanaf de wortel te volgen en steeds de gepaste tak te kiezen tot een bronwaarde corresponderend met een blad verkregen wordt.

Figuur 3.4 schetst hoe een huffmancode C geconstrueerd wordt voor dezelfde toevalsveranderlijke als in de Shannon-Fano-code in figuur 3.3. De verwachte lengte L bedraagt nu 2.35 bps in plaats van 2.45 bps. De redundantie is afgenomen van 0.14 bps naar 0.04 bps.

Optimale code

Een dergelijke huffmancode C is optimaal in de zin dat als C' een willekeurige prefixcode is, dan zal $L_C \leq L_{C'}$. Dit kan recursief bewezen worden



X	$p(x)$	$C(x)$	$l_C(x)$	$C'(x)$
e	0.40	1	1	0
a	0.15	010	3	100
b	0.15	011	3	101
d	0.15	000	3	110
f	0.10	0010	4	1111
c	0.05	0011	4	1110

Figuur 3.4: Voorbeeld van de constructie van een huffmancode $C(x)$ voor dezelfde toevalsveranderlijke als in figuur 3.3. De canonische huffmancode $C'(x)$ is equivalent qua lengte maar kan sneller doorgestuurd en gedecodeerd worden.

door te steunen op de bovenvermelde eigenschappen van een optimale code. Huffmancodes kunnen geen grotere redundantie bereiken dan Shannon-Fanocodes, zodat de grenzen $H \leq L < H + 1$ geldig blijven. Meer nog, Gallager leidde een bovengrens af die meestal strenger is: $H \leq L < H + p_{\max} + 0.086$, waarbij $p_{\max}$ de probabiliteit is van de meest waarschijnlijke waarde [81]. Capocelli *et al.* hebben nog strengere grenzen afgeleid [29, 30]. De ondergrens H wordt bereikt als en slechts als de massafunctie dyadisch is.

Uniciteit

Het algoritme voor de constructie van een huffmancode is niet eenduidig: de sorteringsstap kan aanleiding geven tot verscheidene resultaten. Dit betekent dat huffmancodes kunnen geconstrueerd worden die allen dezelfde verwachte lengte L bereiken, maar die een verschillende verzameling lengten $\{l(x)\}$ genereren. Zo kan een massafunctie met waarden $\{0.4, 0.2, 0.2, 0.1, 0.1\}$ zowel aanleiding geven tot een code met lengten $\{1, 2, 3, 4, 4\}$ als tot een code met lengten $\{2, 2, 2, 3, 3\}$. In deze gevallen gaat de voorkeur uit naar codes die een minimale variantie van de codewoordlengte genereren, of codes met een minimale waarde voor de maximale codewoordlengte en voor de som van de codewoordlengten [219]. Dit komt de stabiliteit van de verwerkingssnelheid tegoed en laat toe deze snelheid meer te optimaliseren. Maar zelfs onder deze criteria zijn de codes niet noodzakelijk uniek bepaald. Indien codewoorden toegekend worden louter op basis van de lengten verkregen uit het huffma-

nalgoritme, ontstaan er huffmancodes die zelf niet via het huffmanalgoritme kunnen geconstrueerd worden.

Canonische huffmancode

Om zowel de opslag van de boom te vereenvoudigen als het decoderen te versnellen, wordt dikwijls gebruikgemaakt van *canonische* huffmancodes. Deze codes voldoen aan volgende eigenschappen: de codewoorden van dezelfde lengte, behorend bij alfabetisch geordende bronwaarden, hebben opeenvolgende waarden indien ze als natuurlijk getal geïnterpreteerd worden en kortere codewoorden stellen lagere getallen voor indien ze geïnterpreteerd worden als binaire fractie. Om een dergelijke code te construeren, wordt het huffmanalgoritme enkel gebruikt om de lengten $\{l(x)\}$ te bepalen. Bovendien moet enkel de verzameling van de codewoordlengten en niet het hele codeboek opgeslagen of doorgestuurd worden. Er bestaan efficiënte methoden om canonische huffmancodes te construeren [93, 227]. Bij wijze van voorbeeld toont figuur 3.4 de resulterende canonische huffmancode $C'(x)$ voor dezelfde toevalsveranderlijke X als in figuur 3.3.

Uitgebreide huffmancode

Een belangrijke beperking van huffmancodes is dat er geen compressie kan verkregen worden indien $\mathcal{X}$ slechts twee waarden bevat. Immers, elk codewoord bestaat uit een geheel aantal codeletters en de kortste codewoorden zijn dan ook '0' en '1.' In dit geval kan $p_{\max}$ zeer groot worden, en kan de redundantie willekeurig dicht tot 1 naderen.

De redundantie kan kleiner worden indien bronwaarden gegroepeerd worden in blokken alvorens ze te coderen. Beschouw daartoe $\mathcal{X}^k$, de verzameling van alle sequenties van lengte k. Een dergelijke sequentie kan geïnterpreteerd worden als een uitkomst van een *uitgebreide* toevalsveranderlijke. Veronderstel nu dat de basistoevalsveranderlijken X_i waaruit een dergelijke uitgebreide toevalsveranderlijke opgebouwd is, onafhankelijk en identiek verdeeld zijn. Definieer verder L_k als de verwachte lengte per bronwaarde, dus $L_k = 1/k \sum_{x_1 \cdots x_k} p(x_1 \cdots x_k) l(x_1 \cdots x_k)$. Indien nu een huffmancode opgesteld wordt voor deze uitgebreide bronwaarden, dan zal $H(X_1 \cdots X_k) \leq kL_k < H(X_1 \cdots X_k) + 1$. Wegens de gemaakte veronderstelling, zal ook $H(X_1 \cdots X_k) = kH(X)$, zodat tenslotte $H(X) \leq L_k < H(X) + 1/k$. Een dergelijke code wordt een *uitgebreide* huffmancode genoemd. Merk op dat hiermee iets anders bedoeld wordt dan de eerder gedefinieerde "uitbreiding" van een huffmancode die niets anders is dan de aaneenrijging van individuele codewoorden.

Het groeperen van bronwaarden in blokken en het toekennen van codewoorden aan die blokken resulteert op deze manier in een code in ruime zin die arbitrair dicht de entropie kan benaderen. De bewering dat huffmancodes optimaal zijn, is dan ook misleidend. Het betekent immers niet dat ze de hoogst mogelijke compressie zullen garanderen, ze zijn enkel optimaal onder de randvoorwaarde dat aan elke bronwaarde een apart codewoord toegekend wordt. De techniek van het groeperen van bronwaarden tot blokken is ook toepasbaar voor andere codes en is een voorbeeld van *alfabetuitbreiding* (*E.* alphabet extension), een algemene term voor de aanpak waarbij overgegaan wordt van een broncode in enge zin naar een broncode in ruime zin. Huffmancodes zijn wel asymptotisch optimaal.

De elegantie van uitgebreide huffmancodes in theorie staat haaks op de moeilijke toepasbaarheid ervan in de praktijk. De codeboeken groeien exponentieel en worden dan ook vlug onhandelbaar groot terwijl de bekomen verlaging van de redundantie beperkt blijft. Een compromis wordt gevonden in de gewijzigde huffmancodes.

Gewijzigde huffmancodes

Indien er veel waarden een zeer kleine waarschijnlijkheid hebben van optreden, dan geeft dit aanleiding tot grote codeboeken waarvan een groot deel slechts weinig gebruikt wordt. Daarom worden de minst waarschijnlijke bronwaarden gegroepeerd tot een subklasse en vervangen door een speciale bronwaarde voorgesteld door het *escapesymbool* ‘`esc`.’ Het opstellen van de huffmancodes verloopt veel efficiënter en het codeboek wordt kleiner. Het codewoord van een dergelijke onwaarschijnlijke bronwaarde bestaat dan uit de aaneenrijging van het codewoord voor ‘`esc`’ en een codewoord om de waarde te onderscheiden binnen deze subklasse. Het tweede codewoord kan dan een elementair codewoord zijn of ook een huffmancodewoord. Een dergelijke code wordt een *gewijzigde* (*E.* modified) huffmancode genoemd [91, 182], en is eveneens een vorm van alfabetuitbreiding. De aanpak waarbij een aantal bronwaarden samengevoegd worden tot één escapesymbool wordt het *escapemechanisme* genoemd, een term die later nog in een andere context opduikt. Ook andere codes dan de huffmancode maken er gebruik van.

Adaptieve huffmancodes

Bij het comprimeren van een gegeven bronsequentie, wordt dikwijls verondersteld dat de massafuncties tijdsafhankelijk zijn. Deze veronderstelling gaat er van uit dat de massafuncties traag variëren want anders kan het bronmodel nooit behoorlijk geschat worden. Het huffmanalgoritme toepassen na elke

bronwaarde geeft aanleiding tot *adaptieve* huffmancodes. Ze hebben twee voordelen: dankzij een adaptieve modellering wordt betere compressie verkregen en ze laten toe om een onbekend bestand in één *beweging* (*E.* pass) in plaats van twee te comprimeren [215]. Het nadeel is dat ze trager zijn, vooral dan wat het decoderen betreft.

Een triviale manier om adaptieve huffmancodes te genereren bestaat erin om voor elke nieuwe bronwaarde een nieuwe huffmanboom te genereren en er het gewenste codewoord uit af te leiden. Dit is de meest algemene aanpak, maar tegelijkertijd ook de aanpak met de hoogste kost qua snelheid. Omdat de massafuncties doorgaans zeer weinig veranderen tussen het coderen van twee opeenvolgende waarden, werden een aantal algoritmen voorgesteld om de huffmanboom dynamisch aan te passen, eerst door Faller [73] en Gallager [81], en later nog verbeterd door Knuth [128] en Vitter [260].

3.4.5 Shannon-Fano-Elias-codes

Elias suggereerde het idee dat later aan de grondslag zou liggen van aritmetische codes; het werd pas later door Abramson gepubliceerd [2].

Stel, zonder aan algemeenheid in te boeten, dat de bronwaarden gegeven worden door natuurlijke getallen $\mathcal{X} = \{1, 2, \ldots, l\}$ en dat verder $p(x) > 0$ voor $x \in \mathcal{X}$ en $p(x) = 0$ voor $x \notin \mathcal{X}$. De cumulatieve distributiefunctie $F(x)$ is gedefinieerd als $F(x) = \sum_{y \leq x} p(y)$. Hierop gebaseerd, wordt de *gewijzigde cumulatieve distributiefunctie* $\bar{F}(x)$ gedefinieerd als $\bar{F}(x) = \sum_{y<x} p(y) + p(x)/2$. Omdat X discreet is, is $F(x)$ een stapfunctie en neemt $\bar{F}(x)$ waarden aan halverwege de stapgrootte.

Omdat $p(x) > 0$, zal $\bar{F}(y) \neq \bar{F}(z)$ als $y \neq z$ met $y, z \in \mathcal{X}$, zodat kennis van $\bar{F}(x)$ toelaat om eenduidig x te bepalen. De binaire voorstelling van $\bar{F}_x$ zou kunnen dienen als codewoord voor x, ware het niet dat $\bar{F}_x$ een oneindig lange binaire voorstelling kan hebben. Het is eenvoudig aan te tonen dat $\lceil -\log p(x) \rceil + 1$ bits ervan volstaan om een prefixcode te bekomen. Deze code wordt de *Shannon-Fano-Elias-code* genoemd.

De bekomen code is doorgaans niet minimaal en dus zeker ook niet optimaal. De verwachte lengte L voldoet aan de grenzen $H \leq L < H(X) + 2$ en ze is minder efficiënt dan Shannon-Fano-codes. De kracht schuilt hierin dat enkel de waarde van $\bar{F}(x)$ en $p(x)$ nodig is om het codewoord te bepalen van een gegeven bronwaarde x. Het is dan ook niet nodig een volledig codeboek op te stellen om slechts één codewoord te kennen.

3.4.6 Aritmetische codes

Basisprincipe

Huffmancodes hebben een aantal nadelen. Ze zijn nutteloos voor binaire bronnen en ze zijn niet efficiënt indien één bronwaarde een zeer grote probabiliteit heeft. Dit kan opgelost worden door uitgebreide huffmancodes te beschouwen, maar die vereisen heel grote codeboeken en de verwachte lengte convergeert maar traag naar de entropie.

Indien de tijdsafhankelijke massafuncties veel veranderen tussen opeenvolgende bronwaarden in een sequentie, kunnen de adaptieve huffmancodes niet meer aangewend worden. Adaptieve contextmodellering bijvoorbeeld zou eisen dat een groot aantal huffmanbomen in parallel onderhouden wordt, of zelfs dat er voor iedere bronwaarde uit een sequentie een nieuwe huffmanboom geconstrueerd wordt.

Beide problemen vinden hun oorsprong in de noodzaak om een hele huffmanboom op te bouwen om het codewoord van één bronwaarde te bepalen. De limiet, waarbij een hele sequentie één lange bronwaarde voorstelt waarvan het codewoord moet bepaald worden, is niet haalbaar in de praktijk.

Shannon-Fano-Elias-codes maken het coderen van een hele sequentie als één enkele typische bronwaarde wel mogelijk. Een waarschijnlijkheidsmodel voor de toevalssequentie $S = X_1 X_2 \cdots X_n$ bepaalt de massafunctie $p(s)$ en de gewijzigde cumulatieve distributiefunctie $\bar{F}(s)$. De binaire voorstelling van $\bar{F}(s)$, beperkt tot $\lceil -\log p(s) \rceil + 1$ bits, vormt tegelijkertijd het codewoord en het bericht. De bovengrens voor de redundantie van het volledige bericht bedraagt 2 bits en de lengte per bronwaarde L zal zeer dicht de entropie van het toevalsproces benaderen: $H \leq L < H + 2/n$, waarbij n de lengte van de sequentie voorstelt.

Het *aritmetisch* coderen van een sequentie $x_1 x_2 \cdots x_n$ van lengte n en met tijdsafhankelijke massafunctie $p_i(x)$ maakt gebruik van het concept van het "huidig waarschijnlijkheidsinterval" $[l_i, h_i[$ op index i. Gebaseerd op het beschreven principe, komt het verwerken van een waarde uit de sequentie neer op het herberekenen van de grenzen van het interval. Op het einde van de sequentie wordt het hele bericht gegenereerd. Een conceptuele implementatie van het basisprincipe verloopt als volgt.

- Initialiseer $[l_0, h_0[= [0, 1[$ en $i = 1$.
- Stel $f_i^{(l)} = \sum_{y<x_i} p_i(y)$ en $f_i^{(h)} = f_i^{(l)} + p_i(x_i)$.
- Verander de waarde van het huidig interval:

$$l_i = l_{i-1} + f_i^{(l)}(h_{i-1} - l_{i-1}),$$

$$h_i = l_{i-1} + f_i^{(h)}(h_{i-1} - l_{i-1}). \tag{3.4}$$

- Verhoog i en herhaal voorgaande stappen tot en met $i = n$.
- Het bericht is de binaire representatie van $(l_i + h_i)/2$, beperkt tot $\lceil -\log(h_i - l_i) \rceil + 1$ bits.

Dit betekent dat de bronwaarden niet moeten gesorteerd worden volgens probabiliteit en dat geen speciale datastructuren vereist zijn. Anderzijds moeten voor het coderen van één codewoord een aantal aritmetische bewerkingen uitgevoerd worden en kan geen gebruik gemaakt worden van een codeboek.

Implementatie

Deze benadering is niet rechtstreeks implementeerbaar omdat reële getallen enkel met eindige precisie kunnen voorgesteld worden. Pasco en Rissanen stelden hiervoor onafhankelijk van elkaar een oplossing voor [171, 190]. Beide voorstellen boden nog geen oplossing voor het feit dat het hele bericht in het geheugen opgeslagen wordt zolang niet de hele sequentie verwerkt is. Incrementele implementaties, die al bits uitbrengen vooraleer het einde van de sequentie bereikt is, werden niet veel later voorgesteld [85, 192, 203]. Een inleidend overzicht werd gepubliceerd door Langdon en Witten beschreef een praktische implementatie die als basis diende voor de huidige implementaties [132, 276]. Een overzicht van de varianten en een beschrijving van een standaardimplementatie is gepubliceerd in [161].

Het beschreven algoritme wordt dan ook op verscheidene manieren aangepast. Er wordt niet met vlottendekommagetallen gewerkt maar met de binaire voorstelling van gehele getallen (doorgaans beperkt tot 32 bit). Het probleem van de eindige precisie die deze voorstelling met zich meebrengt, wordt verholpen door het huidig interval voortdurend te *herschalen* en de voorste bits naar buiten te brengen. Op deze manier verloopt het coderen ook incrementeel.

Het herschalen verdubbelt de lengte van het huidig interval waarbij telkens één bit naar buiten gebracht wordt. Concreet verloopt dit als volgt. Indien het huidig interval $[l_i, h_i[$:

- $\subset [0, 0.5[$: breng een '0' bit buiten en gebruik $[2l_i, 2h_i[$ als waarde voor het nieuwe huidig interval;
- $\subset [0.5, 1[$: breng een '1' bit buiten en gebruik $[2l_i - 1, 2h_i - 1[$ als waarde voor het nieuwe huidig interval;

- $\subset [0.25, 0.75[$: activeer de volgprocedure indien ze nog niet actief is, verhoog de volgteller met 1 en gebruik $[2l_i - 0.5, 2h_i - 0.5[$ als waarde voor het nieuwe huidig interval;

- herhaal deze stappen zolang mogelijk.

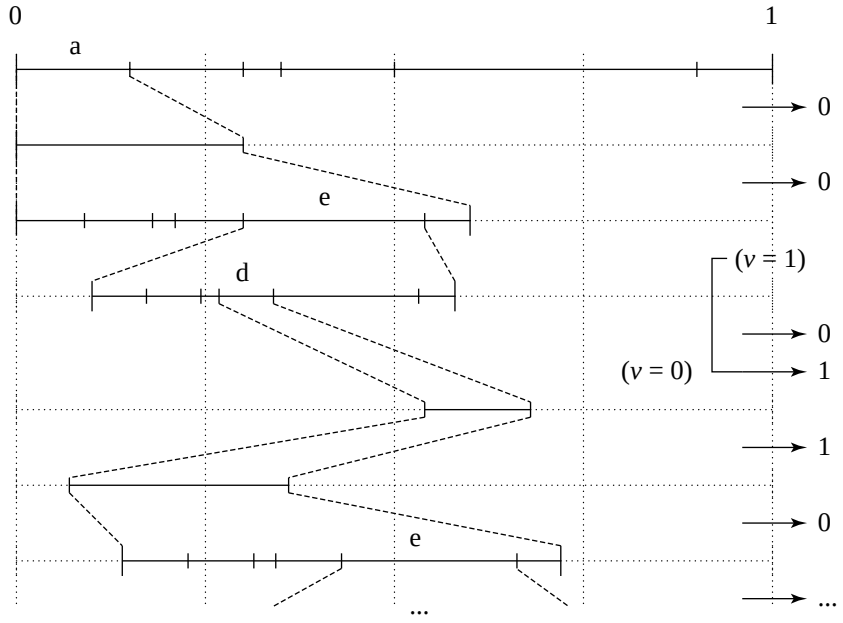
De *volgprocedure* (*E.* follow-on procedure) in het derde geval dient om het probleem te omzeilen dat er nog niet onmiddellijk een bit naar buiten kan gebracht worden [276]. Het interval kan namelijk gecodeerd worden als '$01\cdots$' of als '$10\cdots$', afhankelijk van wat er later volgt. De waarde van de eerstkomende bit is bijgevolg nog niet gekend, maar wat ook de waarde later zal blijken te zijn, de bit die erop volgt heeft de omgekeerde waarde. De volgprocedure markeert deze situatie en telt het aantal bits dat tijdelijk nog niet naar buiten kan gebracht worden. Stel bijvoorbeeld dat later blijkt dat het interval herleid werd tot een interval binnen $[0.5, 0.75[$, dan moet er eerst een '1'-bit uitgeschreven worden, gevolgd door zoveel '0'-bits als aangegeven door de volgteller. Vervolgens moet de volgprocedure weer gedesactiveerd worden en krijgt de volgteller opnieuw de waarde 0. Op deze manier garandeert de herschalingsprocedure dat de lengte van het interval altijd minstens 0.25 bedraagt. Dit genereert een bovengrens voor de afrondingsfout op de cumulatieve probabiliteit indien de onder- en bovengrens van het huidig interval aan eindige precisie voorgesteld worden.

Figuur 3.5 schetst bij wijze van voorbeeld het begin van het coderen van de sequentie '$aede\cdots$', waarbij $\mathcal{X}$ en $p(x)$ dezelfde zijn als in de voorbeelden van een Shannon-Fano-code en een huffmancode. Rechts worden de bits uitgeschreven, één bij iedere herschaling, en v stelt de volgteller voor.

Het decoderen verloopt gelijkaardig aan het coderen. Het waarschijnlijkheidsmodel dient dan ook gekend te zijn en exact hetzelfde als hetgeen gebruikt werd voor het coderen. Er worden altijd voldoende bits ingelezen van het bericht zodat de volgende bronwaarde eenduidig kan gereconstrueerd worden. De aanpassing van de grenzen van het huidig interval evenals het herschalen en het uitschuiven van de voorste bits wordt perfect nagebootst.

Efficiëntie

In theorie is het mogelijk om een aritmetische coder te implementeren die voldoet aan de grenzen $H \leq L < H + 1$ voor de volledige sequentie. In de praktijk zal een goede implementatie hier echter lichtjes van afwijken. Ten eerste omdat een extra symbool moet meegecodeerd worden dat het einde van de sequentie aanduidt. Ten tweede omdat de probabiliteiten moeten afgerond worden zodat ze kunnen voorgesteld worden met eindige precisie. De impact



Figuur 3.5: Voorbeeld van het genereren van een aritmetische code voor de sequentie '$aede\cdots$', voor dezelfde $\mathcal{X}$ en $\{p(x)\}$ als in figuur 3.3 en 3.4. De volgteller v wijst er op dat de volgprocedure ingesteld is.

van het 'einde-sequentie'-symbool is verwaarloosbaar voor voldoend lange sequenties. De toename van het bericht tengevolge van de eindige precisie is van de grootteorde 10^{-4} bits per bronwaarde [276].

Bijzondere implementaties

Het grootste voordeel van aritmetische codes is hun flexibiliteit, hun beperkte geheugenvereiste en hun verwaarloosbare redundantie. Het grootste nadeel is hun lage snelheid die vooral een gevolg is van de aritmetische bewerkingen die voor elke uitgaande bit moeten uitgevoerd worden.

Er zijn dan ook enkele aanpassingen voorgesteld die sneller zijn maar minder efficiënt [162, 163]. Zo bestaan er implementaties die vrij zijn van vermenigvuldigingen [35, 193]. Andere implementaties discretiseren de waarschijnlijkheidsruimte en maken gebruik van transitietabellen [97, 101]; hiervoor wordt vaak de term *quasi-aritmetische* coder gebruikt. Een gedetailleerde beschrijving van aritmetische en quasi-aritmetische codes evenals een

efficiënte implementatie is te vinden in [104].

Dikwijls zijn de bronwaarden binair en geoptimaliseerde algoritmen maken hier dankbaar gebruik van. Door benaderingen van de probabiliteiten in te voeren, verloopt de codering veel sneller terwijl de efficiëntie slechts weinig afneemt. De *scheefcoder* (*E.* skew coder) benadert de waarschijnlijkheid van de minst waarschijnlijke waarde door een negatieve macht van 2 en maakt enkel gebruik van schuifoperaties en optellingen [137]. Hieruit volgde later de *Q-coder* die de waarschijnlijkheden op iets hogere waarden kwantiseert en gebruik maakt van een opzoektabel [157, 173]. Het is een coder die de waarschijnlijkheid van het minst waarschijnlijke symbool (*E.* least probable symbol) of LPS onderbrengt in één van 30 voorgedefinieerde intervallen. Deze coder is geschikt voor zowel software- als hardware-implementatie.

De *QM-coder* is een verfijning van de Q-coder die vooral gericht is op software-implementatie. Deze coder is speciaal ontworpen voor een aantal recente compressiestandaarden [118,119,172]. Het aantal waarschijnlijkheidsintervallen bedraagt nu 113 en is samen met de waarde van het *meest waarschijnlijke symbool* (*E.* most probable symbol) of MPS voor te stellen in één byte (er zijn dus effectief 226 verschillende toestanden). De *Qx-coder* integreert beide implementaties in één hardwareverpakking [230]. Tenslotte vermelden we nog de *MQ-coder*, een variant van de *QM-coder* die in een aantal pas verschenen standaarden gebruikt wordt [120, 121].

Bereikcode

De octrooien die rusten op aritmetische codes hebben hun populariteit sterk ingeperkt, al zijn een aantal ervan intussen al vervallen. *Bereikcodering* (*E.* range encoding) is vrij van octrooien en werkt gelijkaardig. Het onderscheid is dat de uitvoer gegroepeerd wordt in bytes in plaats van bits. Dit resulteert in een minder optimale lengte (circa 0.01%) maar de snelheid verdubbelt. Bereikcodes zijn ouder dan aritmetische codes, maar doken recent opnieuw op [144, 217].

3.4.7 Tunstallcodes

De hierboven beschreven entropiecodes zijn allemaal blok-variabel-codes. Nagenoeg alle recente praktische compressieschema's maken er gebruik van. *Tunstallcodes* zijn een eenvoudig voorbeeld van variabel-blok-codes [246] die minstens even efficiënt zijn.

Is $l = l(\mathcal{X})$, dan kan een complete en eenduidige tunstallcode geconstrueerd worden met $v = l + k(l-1)$ codewoorden van (min of meer) constante lengte, waarbij k een natuurlijk getal is. Indien $\mathcal{X}$ voortgebracht wordt door een discrete geheugenloze bron, convergeert de verwachte lengte per symbool

naar de entropie H voor $v \to \infty$. Met elke keuze voor k komt een waarde voor v overeen en kan een tunstallcode opgebouwd worden. De subsequenties hebben een variabele lengte maar de lengte van de codewoorden is vast en bedraagt precies $\lceil \log v \rceil$.

Het opbouwen van een code voor een gegeven waarde van v en een gegeven bronverzameling $\mathcal{X}$ gebeurt als volgt. Stelt $\mathcal{X}_C^*$ de verzameling van alle subsequenties van het codeboek voor, dan wordt deze verzameling initieel gelijkgesteld aan $\mathcal{X}$. Dit betekent dat initieel enkel subsequenties van lengte één toegelaten zijn en het voorlopige codeboek telt precies l van deze atomaire subsequenties. Vervolgens komt de iteratiestap: is s de meest waarschijnlijke subsequentie van dit voorlopige codeboek, dan wordt s vervangen door alle mogelijke uitbreidingen sx van s met $x \in \mathcal{X}$. Symbolisch uitgedrukt betekent dit dat $\mathcal{X}_C^*$ vervangen wordt door $\mathcal{X}_C^* \setminus \{s\} \cup \{sx : x \in \mathcal{X}\}$. Het aantal subsequenties in het voorlopige codeboek neemt toe met $l - 1$ nieuwe subsequenties. Deze iteratiestap herhaalt zich precies k keer waardoor het voorlopige codeboek v subsequenties bevat. De resulterende subsequenties van variabele lengte worden lexicografisch geordend en aan elke subsequentie wordt een index toegekend. Het geassocieerde codewoord wordt gevormd door de elementaire binaire voorstelling van de respectieve index.

Qua complexiteit en efficiëntie zijn tunstallcodes vergelijkbaar met huffmancodes, maar er zijn enkele subtiele verschillen. De decodeersnelheid zal meestal hoger liggen bij tunstallcodes en de codegeneratie verloopt eenvoudiger. Het adaptief maken van de code resulteert voor beide in een veel ingewikkeldere codeerstap en de grootte van de sequentiebomen neemt exponentieel toe zodat de convergentie naar optimale lengte traag verloopt en in de praktijk moeilijk haalbaar is. Het optimaliseren van tunstallcodes voor bronnen met geheugen (markovbronnen) is nog volop in onderzoek [117, 214, 243].

3.5 Codes voor de natuurlijke getallen

Veel entropiecodes kunnen enkel geconstrueerd worden voor een eindig aantal bronwaarden waarvan de probabiliteiten gekend zijn. Coderen vergt doorgaans een niet geringe hoeveelheid rekenwerk en decoderen maakt dikwijls gebruik van een aanzienlijke hoeveelheid geheugen om het codeboek op te slaan. Soms is het wenselijk over een code te beschikken die snel is, weinig geheugen gebruikt en toelaat om alle natuurlijke getallen voor te stellen. Dergelijke *onbegrensde* codes kunnen voor diverse uiteenlopende toepassingen gebruikt worden. Dikwijls betreffen ze een hele klasse van codes die van elkaar onderscheiden worden door een parameter. Het zijn geen zuivere entropiecodes omdat ze voor de constructie van de code geen gebruik maken van de absolute

grootte van de waarschijnlijkheidsmassa's, maar enkel van de relatieve grootte ervan.

De bronwaarden worden gesorteerd volgens dalende probabiliteit en de codewoorden worden toegekend. Een code is *universeel* indien de verwachte lengte per codewoord begrensd wordt door $c_1(H+c_2)$, met c_1 en c_2 constanten. Indien $c_1 = 1$ dan is de code ook asymptotisch optimaal.

De probabiliteiten van de bronwaarden moeten dan ook niet absoluut gekend zijn, het volstaat indien de bronwaarden kunnen gesorteerd worden volgens afnemende probabiliteit. Bovendien zijn de codewoorden vast en is het mogelijk elk codewoord afzonderlijk te berekenen. Coderen en decoderen verloopt dan ook snel en vergt weinig geheugen. Daar staat tegenover dat de verwachte lengte per codewoord nagenoeg nooit zo klein zal zijn als bij huffmancodes, zelfs voor asymptotisch optimale universele codes. Uitzondering hierop vormen probabiliteitsdistributies waarvoor de huffmancodes nagenoeg dezelfde lengten opleveren.

Hierna wordt met "de binaire code" van een natuurlijk getal x de klassieke binaire voorstelling bedoeld. Dit stemt overeen met het codewoord $C(n)$ van de elementaire binaire code van $\mathcal{X}_x = \{0, 1, \ldots, 2^{\lceil \log_2 x \rceil} - 1\}$. Zoals eerder vermeld is deze klassieke binaire voorstelling van de natuurlijke getallen wel regulier maar niet uniek decodeerbaar en bijgevolg voor veel toepassingen nutteloos als code. Merk op dat voor sommige codes bij conventie de natuurlijke getallen beginnen bij 1, maar voor andere bij 0.

3.5.1 Unaire code

Een unaire code drukt als het ware de natuurlijke getallen uit in een eentallig stelsel met een stopbit. Een natuurlijk getal $x > 0$ wordt voorgesteld door $x - 1$ keer een '1', gevolgd door een '0', of omgekeerd. Een unaire code is niet universeel, maar wordt veel gebruikt om universele codes op te bouwen. Code C_α in tabel 3.4 geeft een unaire code weer van de eerste acht natuurlijke getallen.

3.5.2 Eliascodes

Een aantal universele codes werd voorgesteld door Elias [69]. Indien vooral kleine getallen optreden, kan de *gammacode* gebruikt worden. De codewoorden bestaan uit een alternering van de bits van een unaire en de binaire code. Code C'_γ in tabel 3.4 geeft de opeenvolging van de unaire en de binaire code en code C_γ geeft de gammacode die uit C'_γ bekomen wordt door de bits van beide subcodewoorden te *alterneren* (*E.* interleave). Indien daarentegen ook

Tabel 3.4: Codes voor de natuurlijke getallen.

n	C_α	C'_γ	C_γ	C_δ	G_2	G_3
0					0 0	0 0
1	0	1	1	1	0 1	0 10
2	10	01 0	001	001 0	10 0	0 11
3	110	01 1	011	001 1	10 1	10 0
4	1110	001 00	00001	011 00	110 0	10 10
5	11110	001 01	00011	011 01	110 1	10 11
6	111110	001 10	01001	011 10	1110 0	110 0
7	1111110	001 11	01011	011 11	1110 1	110 10
8	11111110	0001 000	0000001	00001 000	11110 0	110 11

veel grote getallen voorkomen, is de *deltacode*, een uitbreiding van de gammacode, efficiënter. Elk codewoord bestaat uit de gammacode van de lengte van de binaire voorstelling van het getal, gevolgd door de binaire voorstelling zelf, zie code C_δ. Beide codes zijn universeel, maar enkel de deltacode is asymptotisch optimaal.

3.5.3 Golomb- en ricecodes

De *golombcode* hangt af van een natuurlijke parameter b, ook wel het *grondtal* (*E.* base) genoemd [83]. Het codewoord voor een natuurlijk getal $x \geq 0$ wordt opgebouwd uit twee delen. Het eerste deel wordt gevormd door de unaire voorstelling van $q = x \div b + 1$, waarbij $x \div b$ de gehele deling $\lfloor x/b \rfloor$ definieert. Dit wordt aangevuld met het tweede deel dat bepaald wordt door de binaire voorstelling van de rest $r = x \bmod b$. Als $b = 2^k$, met k een natuurlijk getal, wordt de rest r voorgesteld in precies k bits. De binaire voorstelling van r is eenduidig qua lengte en een dergelijke code wordt een *ricecode* met parameter k genoemd [184]. In de andere gevallen wordt de rest r voorgesteld in $\lfloor k \rfloor$ of $\lceil k \rceil$ bits, overeenkomstig de geleidelijke binaire voorstelling.

Golombcodes zijn optimaal indien de probabiliteitsdistributie geometrisch is (exponentieel afnemend). In dat geval stemmen ze overeen met een huffmancode. Code G_2 en G_3 in tabel 3.4 stemmen respectievelijk overeen met de golombcode met parameter 2 en 3.

3.5.4 Start-stap-stop-codes

Aan de hand van drie parameters ($s = start$, $k = stap$ en $t = stop$) wordt een grote verzameling codes gedefinieerd die *start-stap-stop-codes* genoemd worden [77]. De codewoorden bestaan uit alle mogelijke binaire codes van lengte $l = s + jk$ met j een natuurlijk getal en waarvoor $l \leq t$, voorafgegaan door de unaire code van l (indien $l = t$ wordt de laatste bit van de unaire code weggelaten). Indien de parameters eindig zijn, bevat de code ook een eindig aantal codewoorden. Enkele van de hierboven beschreven codes behoren ook tot deze klasse. Zo genereren de parameters $(0, 1, \infty)$ de gammacode op een herordening van de bits na.

3.6 Samenvatting

In dit hoofdstuk hebben we gedefinieerd wat een code is, en welke eigenschappen relevant zijn voor compressie. De entropie bepaalt de ondergrens voor de verwachte lengte die haalbaar is met om het even welke uniek decodeerbare code. Entropiecodes zijn codes die rekening houden met de waarschijnlijkheid van optreden van elke bronwaarde. Maar naast de efficiëntie en het al dan niet optimale karakter van de code, zijn er nog een aantal andere belangrijke eigenschappen zoals de snelheid, de eenvoud van implementatie, de mate van adaptiviteit, geheugenvereisten en de mogelijkheid om een oneindige bronverzameling te coderen.

Huffmancodes zijn optimale entropiecodes onder de randvoorwaarde dat met elke bronwaarde een codewoord van gehele lengte geassocieerd wordt. De constructie verloopt snel en varianten zoals uitgebreide, gewijzigde en adaptieve huffmancodes bieden een passend antwoord op de meeste nadelen. Enkel voor binaire bronsequenties zijn huffmancodes geen elegante oplossing.

Aritmetische codes kennen geen van deze tekortkomingen. Door exact één codewoord te genereren voor de gehele sequentie, zijn ze in theorie optimaal. De grote nadelen zijn de lage verwerkingssnelheid en het waas van octrooien die deze techniek omhullen. Om die redenen verschijnen ook hier steeds meer varianten als de QM-, de Qx- en de MQ-coder, die erop gericht zijn de snelheid te verhogen en de octrooibescherming te omzeilen.

Tot slot zijn er nog een aantal geparametriseerde codes die gedefinieerd zijn over de natuurlijke getallen. Ze zijn optimaal voor bepaalde voorgedefinieerde distributies. Codes zoals de golomb- en ricecodes zijn heel snel en kunnen via hun parameter alsnog een hoge graad van adaptiviteit bekomen.

Hoofdstuk 4

State of the art van verliesloze beeldcompressie

4.1 Inleiding

De voorbije decennia zijn in de literatuur veel nieuwe algoritmen voorgesteld voor de verliesloze compressie van beelden. Ze onderscheiden zich van elkaar qua beeldvoorstelling, onderliggend model, data-afhankelijkheid, complexiteit, snelheid en dies meer. Toch zijn er ook een groot aantal parallellen tussen deze methoden. In het kader van dit werk is het onmogelijk exhaustief al deze methoden in detail te bespreken en in dit hoofdstuk beperken we ons dan ook tot een representatief aantal van hen. Voor enkele van deze methoden geven we de experimentele compressieresultaten op een aantal relevante testbeelden uit verschillende domeinen.

Voor de meeste van deze algoritmen stond bij het ontwerp een specifieke beeldklasse voor ogen: binaire beelden, monochrome beelden, kleurenbeelden, driedimensionale beeldensets en meercomponentenbeelden. Daarom worden de methoden in dit hoofdstuk in eerste instantie volgens toepassingsgebied gegroepeerd. Binnen elk toepassingsgebied worden ze vervolgens onderverdeeld volgens data-afhankelijkheid, omdat dit zowel de beelddecorrelatie als de opbouw van de visuele reconstructie bepaalt. Een aantal basiswerken over verliesloze compressie van tekst en beelden beschrijven deze technieken in meer detail [14, 37, 82, 87, 95, 165, 166, 182, 209, 215, 233, 273].

Niettegenstaande de eigen publicaties over de state of the art dateren van voor 1998, is er een inspanning geleverd om de lijst van gepubliceerde compressietechnieken in dit proefschrift up-to-date te houden. Zo komen hier heel recente standaarden als JPEG2000 en JBIG2, beiden verschenen in 2001, uitvoerig aan bod. Deze state of the art moet dan ook geïnterpreteerd worden als

een state of the art anno eind 2001. In dit opzicht bevat dit hoofdstuk technieken die meer recent zijn dan de eigen bijdragen in de volgende hoofdstukken. De opbouw van het proefschrift volgt bijgevolg geen chronologische lijn. Het is belangrijk deze wisselende tijdslijn in overweging te nemen bij het doornemen van het proefschrift.

In dit hoofdstuk laten we de technische details van de beschreven algoritmen grotendeels achterwege en besteden we vooral aandacht aan de onderliggende principes. Paragraaf 4.2 beschrijft de basisprincipes van enkele populaire compressiealgoritmen voor algemene bronsequenties. Al zijn ze niet specifiek ontworpen voor beeldcompressie, toch kunnen ze met wisselend succes toegepast worden en vormen ze veelal de basis van de beeldcompressietechnieken. Paragraaf 4.3 behandelt enkele verliesloze compressiealgoritmen die specifiek voor binaire beelden ontworpen werden. In paragraaf 4.4 worden algoritmen voor monochrome beelden (grijswaardenbeelden of de individuele componenten van kleurenbeelden) besproken. Uitbreidingen naar meer kleuren en hogere dimensies komen aan bod in paragraafs 4.5 en 4.6. Enkele van de besproken methoden worden vervolgens toegepast op een aantal standaardtestbeelden en beelden uit de drukvoorbereidingsindustrie en de medische wereld. De resultaten op het vlak van efficiëntie en verwerkingssnelheid worden voorgesteld in paragraaf 4.7. Hierop voortbouwend worden enkele eenvoudige aanpassingen voor de drukvoorbereidingsindustrie voorgesteld in paragraaf 4.8. Tenslotte eindigt paragraaf 4.9 met een samenvatting en worden de eigen bijdragen opgesomd.

4.2 Algemene methoden

Historisch gezien zijn de eerste compressiemethoden ontworpen met tekstcompressie als toepassing voor ogen. Omdat tekst nagenoeg altijd verliesloos gecomprimeerd wordt, stammen de meeste methoden voor verliesloze beeldcompressie dan ook uit dit domein. De ontwikkelde modellen zijn gevorderd en dermate universeel dat ze met beperkt succes ook voor beeldcompressie gebruikt kunnen worden.

Het beschouwen van tweedimensionale beeldinformatie als een eendimensionaal tekstbestand verwaarloost enerzijds onmiddellijk enkele typische eigenschappen van beelden. Ten eerste wordt het tweedimensionale karakter van de beeldinformatie niet langer in rekening gebracht. Ten tweede vallen de pixelgrenzen van een beeld, dit zijn de aflijningen van de pixelintensiteit in de binaire voorstelling, niet altijd samen met de karaktergrenzen van een tekst. De pixelgrenzen worden immers gekenmerkt door een periode die varieert van 1 bit voor binaire beelden over 8 bit voor normale grijswaardenbeelden en

24 bit voor RGB-beelden tot 32 bit voor CMYK-beelden, terwijl de karaktergrenzen van tekst nagenoeg altijd door een periode van 8 bit gekenmerkt zijn. Ten derde wordt de relatieve grootte van de pixelintensiteiten buiten beschouwing gelaten: een meer waarschijnlijke overgang van intensiteit 100 naar 101 wordt op gelijke basis behandeld als een minder waarschijnlijke overgang van intensiteit 100 naar 200. Ten vierde worden aspecten als kleurinformatie, een derde spatiale dimensie of een tijdsdimensie evenmin in rekening gebracht.

Anderzijds zijn deze methoden dikwijls zeer krachtig, universeel inzetbaar, soms bijzonder populair in gebruik en er bestaan geoptimaliseerde implementaties voor. Bovendien vormt de onderliggende modellering ervan de basis voor de algoritmen die specifiek gericht zijn op de verliesloze compressie van beelden.

Op het gebied van data-afhankelijkheid maken we een onderscheid tussen *karaktergebaseerde* en *blokgebaseerde* methoden. Voor de karaktergebaseerde methoden manifesteert de wisselwerking tussen decorrelatie en variabelelengtecodering zich na elk karakter, terwijl dit zich voor blokgebaseerde methoden na één of meerdere karakters voordoet. Deze laatsten kunnen met blokken van veranderlijke of constante lengte werken. De term *karakter* verwijst hier algemeen naar een afzonderlijke bronwaarde, zoals een letterteken in een tekst of een pixelwaarde in een beeld. Karaktergebaseerde methoden zijn doorgaans trager maar efficiënter dan blokgebaseerde.

4.2.1 Karaktergebaseerde methoden — het contextmodel

De *karaktergebaseerde* methoden verwerken de sequentie karakter per karakter (meestal byte per byte). Ieder nieuw karakter wordt beschouwd als een resultaat van een toevalsexperiment waarvoor een probabiliteit van optreden gekend is of geschat kan worden. Hiervoor is een waarschijnlijkheidsmodel nodig en de meeste methoden onderscheiden zich van elkaar door een ander waarschijnlijkheidsmodel te veronderstellen.

Het schatten van de probabiliteit $p(x)$ van elke mogelijke uitkomst $x \in \mathcal{X}$ gebeurt aan de hand van statistieken opgebouwd uit het eigenlijke tekstbestand en daarom worden deze methoden *statistisch* genoemd. Typisch zijn deze statistieken gelijk aan de relatieve frequentie van voorkomen, dus $\hat{p}(x) = c(x)/\sum_{t\in\mathcal{X}} c(t)$, waarbij de notatie $\hat{p}(x)$ duidt op een schatting voor de ongekende $p(x)$ en waarbij $c(\cdot)$ aangeeft hoeveel keer '$\cdot$' optreedt. De datastructuur die deze frequentiegebaseerde statistieken voorstelt is meestal een tabel en wordt de *frequentietabel* genoemd. Het is gebruikelijk om de absolute frequentie $c(x)$ van optreden, ook de *telwaarde* (*E.* count) genoemd, op te slaan en de relatieve frequentie te berekenen indien die nodig is.

Voor verliesloze compressie is het noodzakelijk dat de decoder over exact dezelfde statistieken kan beschikken als de coder en soms moet daarvoor *zij-informatie* (*E.* side information) ongecomprimeerd doorgestuurd worden. Bij de machinevoorstelling van de statistieken dient rekening gehouden te worden met de eindige precisie. Een aritmetische code is de aangewezen manier om de geschatte probabiliteiten om te zetten in een minimale hoeveelheid gecodeerde bits.

Statisch en adaptief model

Een statistisch model kan verschillende gradaties tonen in het vermogen om zich aan te passen aan de statistische eigenschappen van de sequentie:

- Een *statisch* model gebruikt statistieken die geoptimaliseerd zijn voor een verzameling typische sequenties en niet telkens aangepast worden aan elke nieuwe sequentie.

- Een *semi-adaptief* model bouwt gedurende een eerste beweging statistieken op voor de gehele sequentie en gebruikt die gedurende een tweede beweging om de sequentie te coderen. Gedurende deze codeerbeweging wijzigen de statistieken niet meer. Deze statistieken dienen als zij-informatie opgeslaan of doorgestuurd te worden naar de decoder om verliesloze reconstructie te garanderen. Dikwijls is het niet nodig alle statistieken door te sturen, maar volstaat de code die eruit afgeleid werd. Deze methode verwerkt de sequentie twee keer en is bijgevolg ongeschikt voor gepijplijnde verwerking van de informatie.

- Een *adaptief* model begint met een initieel (eventueel leeg) model en past zich tijdens het coderen voortdurend aan aan de reeds verwerkte gegevens. De decoder bootst het gedrag van de coder na en het is niet nodig om zij-informatie door te sturen. Deze aanpak is optimaal indien de sequentie niet-stationair is.

Overgaan van een statisch model via een semi-adaptief naar een adaptief model brengt meestal een hogere compressieverhouding en een lagere verwerkingssnelheid met zich mee. Bepaalde bronnen brengen statische en semi-adaptieve modellen samen onder de term *niet-adaptieve* modellen [189].

De initialisatie en herschaling van een adaptief model moet rekening houden met het *nulfrequentieprobleem* (*E.* zero-frequency problem). Wordt de relatieve frequentie van optreden $c(x)/\sum_{t\in\mathcal{X}} c(t)$ geschat op basis van de reeds verwerkte waarden, dan levert dit een geschatte probabiliteit $\hat{p}(x) = 0$ op

Tabel 4.1: Orde-0 relatievefrequentietabel voor Nederlandstalige tekst waarbij de waarden zijn uitgedrukt in procent. Deze schattingen zijn gebaseerd op een elementaire statistische verwerking van een uitgebreide woordenlijst.

'*a*'	'*b*'	'*c*'	'*d*'	'*e*'	'*f*'	$\cdots$	'*z*'
6.94	2.19	2.45	3.72	17.07	1.58	$\cdots$	0.73

voor het eerste optreden van een x. Er zijn bijgevolg $-\log 0 = +\infty$ bits nodig om deze uitkomst te coderen. Omdat de frequenties met eindige precisie opgeslagen worden, kan dit probleem ook optreden bij herschaling van de frequentietabel. In het eerste geval kan dit probleem vermeden worden door de frequentietabel te initialiseren met van 0 verschillende waarden. In het tweede geval biedt een gepaste afronding een oplossing.

Orde-0 model

Een eenvoudig statistisch model houdt rekening met de frequentie van voorkomen van elk karakter op zich, ongeacht welke karakters eraan voorafgingen of welke karakters erop volgden. Omdat de omgeving niet in rekening gebracht wordt, spreekt men van een *orde-0 model*. Als schatting voor de probabiliteit van x wordt $\hat{p}(x) = c(x)/\sum_{t\in\mathcal{X}} c(t)$ genomen. De interpretatie van $c(\cdot)$ is afhankelijk van de adaptiviteit van het model. De frequentietabel bevat slechts één rij en heeft $l-1$ vrijheidsgraden, waarbij $l = l(\mathcal{X})$.

In het geval van Nederlandstalige tekst bijvoorbeeld, komt de letter '*q*' veel minder voor dan de letter '*e*' en dit wordt uitgebuit om een tekst te comprimeren. Tabel 4.1 geeft een typische relatievefrequentietabel weer voor Nederlandstalige tekst. Het vermelden waard in dit kader zijn *lipogrammen*, teksten waarin met opzet een bepaalde letter weggelaten is. Een sprekend voorbeeld hiervan is "Gadsby", een Engelstalige novelle uit 1939 van E. V. Wright die meer dan 50 000 woorden telt maar de letter '*e*' niet bevat.

Contextmodel

Een belangrijke eigenschap van de entropie $H(X)$ van een toevalsveranderlijke X is dat ze afneemt indien X geconditioneerd wordt op een toevalsveranderlijke Y die niet onafhankelijk is van X. Anders geformuleerd, $H(X|Y) \leq H(X)$ met gelijkheid enkel indien X en Y onafhankelijk zijn van elkaar.

In het geval van tekst zijn de opeenvolgende karakters niet onafhankelijk en het conditioneren van een karakter op het voorgaande karakter zorgt voor

een afname van de entropie. Zo volgt op de letter 'q' nagenoeg altijd de letter 'u', zodat de geschatte massafunctie $\hat{p}(\cdot|\text{'}q\text{'})$ waarvoor 'q' als vorig karakter optrad een grote piek toont voor de letter 'u'.

Dit vormt de basis voor het succes van een *contextmodel*: de schatting voor de probabiliteit van een gegeven uitkomst wordt bepaald door de relatieve frequentie van optreden onder een identieke voorgeschiedenis. Een *contextfunctie* beeldt de voorgeschiedenis af op een unieke *conditionerende klasse* of *context*. Een *eindigecontextmodel* (*E.* finite context model) of FCM heeft een contextfunctie die slechts een eindig interval van de voorgeschiedenis als argument heeft. Als schatting voor de probabiliteit van x onder de context y kan $\hat{p}(x|y) = c(yx)/\sum_{t\in\mathcal{X}} c(yt)$ genomen worden. In het kader van verliesloze compressie is het belangrijk enkel te conditioneren op toevalsveranderlijken die al gekend zijn op basis van de voorgeschiedenis en niet op basis van de toekomst. Rissanen beschrijft de formalisering van een veralgemeend contextmodel [189]. Een contextmodel veronderstelt dat het onderliggende toevalsproces markoviaans is.

Een *klassiek contextmodel van orde* q gebruikt de aaneenrijging van de voorbije q bronwaarden als context. De *orde* verwijst hier dus naar het aantal voorgaande bronwaarden dat in rekening gebracht wordt. Er zijn contextmodellen met *vaste orde* en *variabele orde*. Bij uitbreiding wordt de term *orde-*(-1) *model* gedefinieerd als een model waar geen statistische informatie gebruikt wordt en de massafunctie uniform verondersteld is. Zoals reeds eerder vermeld gebruikt een orde-0 model statistieken zonder contexten. Een klassiek contextmodel van orde q kan beschouwd worden als een aggregaat van l^q orde-0 modellen die elk behoren bij een specifieke context. Een contextmodel is bijgevolg een voorbeeld van een *samengesteldebronmodel* (*E.* composite source model).

Tabel 4.2 toont een frequentietabel van een orde-1 contextmodel voor tekstcompressie. De eerste kolom somt alle mogelijke waarden voor Y op en iedere rij geeft de geschatte massafunctie van X weer voor één context y.

Contextverdunning

Een klassiek contextmodel van orde q voor een bronalfabet met l verschillende karakters wordt beschreven aan de hand van een frequentietabel met l^{q+1} frequenties. Omdat voor een gegeven context enkel de relatieve frequenties van belang zijn, kent dit model $l^q(l-1)$ vrijheidsgraden. Anderzijds neemt de entropie van een vaste-orde-model af bij toenemende q zolang de bijkomende contextveranderlijken niet onafhankelijk zijn. Bij een goede keuze van de contextveranderlijken zal deze afname groot zijn bij lage ordes en klein bij hogere

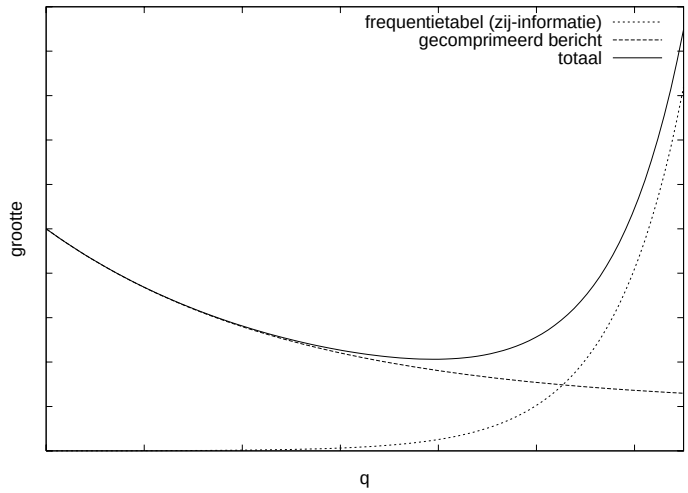
Tabel 4.2: Orde-1 absolutefrequentietabel voor Nederlandstalige tekst. De massafunctie met '*q*' als context vertoont duidelijk een piek bij '*u*'.

$Y \setminus X$	$\cdots$	'*s*'	'*t*'	'*u*'	'*v*'	'*w*'	$\cdots$
$\cdots$				$\cdots$			
'*p*'	$\cdots$	232	399	269	46	21	$\cdots$
'*q*'	$\cdots$	0	0	168	0	2	$\cdots$
'*r*'	$\cdots$	1972	1827	1215	253	245	$\cdots$
$\cdots$				$\cdots$			

ordes. De exponentiële toename van de grootte van het model bij toenemende orde staat echter haaks op deze langzame afname van de entropie. Er is steeds meer geheugen en rekenkracht nodig terwijl de efficiëntie steeds trager toeneemt. Daarenboven zal ook de verwerkingssnelheid van het algoritme afnemen. Het kwalitatieve gedrag van de efficiëntie als functie van de orde is gelijkaardig voor semi-adaptieve en adaptieve modellen.

In het geval van een semi-adaptief model dient de frequentietabel doorgestuurd te worden als zij-informatie. Voor lage ordes is deze overhead verwaarloosbaar, maar voor toenemende ordes kan ze de grootte van de entropiegecodeerde stroom overtreffen omdat alle bronsequenties eindig zijn. De compressieverhouding als functie van de orde vertoont een maximum; de plaats van dit maximum hangt af van de grootte van het gecomprimeerde bestand. Ter illustratie schetst figuur 4.1 de grootte van de frequentietabel (exponentieel stijgend) en van het gecomprimeerd bestand (monotoon dalend) bij toenemende orde. De som van beide vertoont een minimum, wat correspondeert met een maximum voor de compressieverhouding. De mogelijkheid om de frequentietabel ook te comprimeren wordt hier buiten beschouwing gelaten.

In het geval van een adaptief model moet voor elke context een massafunctie geschat worden. Maar om het nulfrequentieprobleem te vermijden wordt elke massafunctie geïnitialiseerd met een uniforme massafunctie en convergeert deze traag naar de ware massafunctie. Indien onvoldoende trainingswaarden beschikbaar zijn, resulteert dit in een slechte modellering en een lage efficiëntie. Dit probleem wordt *contextverdunning* (*E.* context dilution) genoemd: er zijn te weinig waarden aanwezig om voor elke context een adequate schatting van de statistieken te maken. Ook hier vertoont de compressieverhouding een maximum als functie van de orde. Het optreden van contextverdunning is in het theoretisch kader van stochastische complexiteit gekend als het zich manifesteren van de *modelkost* (*E.* model cost) [186]. De modelkost



Figuur 4.1: Verloop van de grootte van de frequentietabel en het gecomprimeerd bericht bij toenemende orde q. De curve vertoont een minimum.

is een maat voor de vertraging van de convergentie als gevolg van het groter aantal vrijheidsgraden in de modelvorming.

Gemengde modellen

De modellering verbeteren zonder contextverdunning te veroorzaken is mogelijk door contextmodellen van verschillende orde te mengen. In een *variabele-orde-model* of *gemengd* model (*E.* blended model) worden frequentieschattingen voor zowel hoge als lage ordes gecombineerd wat leidt tot een gemengde schatting. Dikwijls kan geen relevante frequentieschatting gemaakt worden voor hoge ordes en onder die omstandigheden komen enkel de schattingen voor lage ordes in aanmerking bij het mengen. Op deze manier wordt de gemengde schatting bepaald door de hoge ordes indien deze een relevante schatting genereren en door de lage ordes in de andere gevallen. Het probleem van contextverdunning wordt op een deterministische manier vermeden. De formulering *voorspelling door partiële overeenkomst* (*E.* prediction by partial match) of PPM wijst erop dat de overeenkomst tussen eindige voorgeschiedenis en context niet volledig moet zijn [39].

Een gemengd model combineert modellen van orde -1 tot en met een maximale orde q. Per definitie kent het orde-(-1) model een gelijke en van nul verschillende probabiliteit toe aan alle symbolen. Dit model wordt gebruikt omdat het als eigenschap heeft dat het nulfrequentieprobleem niet kan

optreden. De vele frequentietabellen behorend bij hogere ordes zijn meestal ijl terwijl de weinige frequentietabellen voor lagere ordes dik bevolkt zijn. De statistieken worden gezamenlijk geïmplementeerd als een *trie* (van 're-*trie*-val'), zie figuur 4.2. Een trie is een boomstructuur (*E.* tree) waarbij het label van elk kind de aaneenrijging is van het label van de ouder en een karakter dat dit kind onderscheidt van de andere kinderen. De afmeting van de trie is niet langer van de vorm l^{q+1} maar is op een constante factor na beperkt door de grootte van de bronsequentie.

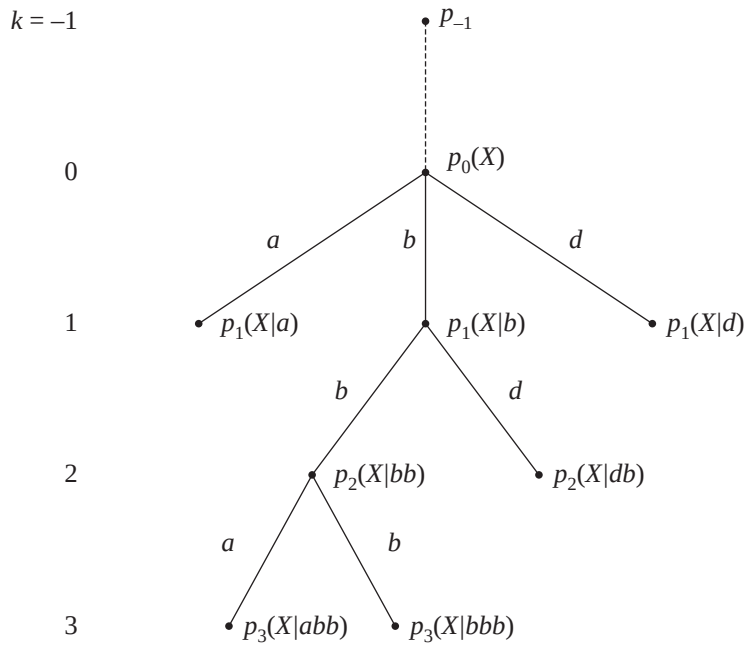
Stelt $s = x_1 x_2 \cdots x_n$ de totale sequentie voor van lengte n, dan wordt op tijdstip i de bronwaarde x_i verwerkt en is de voorgeschiedenis $y = x_1 x_2 \cdots x_{i-1}$. Hierna gebruiken we de verkorte notatie $x = x_i$ voor de bronwaarde en $y^k = x_{i-k} \cdots x_{i-2} x_{i-1} = x_{i-1}^k$ voor de huidige context van orde k. Bij iedere orde $k \in \{-1, \ldots, q\}$ hoort een geschatte probabiliteit $\hat{p}_k(x|y^k)$. De gemengde probabiliteit wordt geschat als

$$\hat{p}(x|y) = \hat{p}(x|y^q) = \sum_{k=-1}^{q} w_k(y^q)\, \hat{p}_k(x|y^k),$$

waarbij de gewichten w_k genormaliseerd zijn zodat $\sum_{k=-1}^{q} w_k(y^q) = 1$ voor alle y. Implementaties van een gemengd contextmodel onderscheiden zich in de toekenning van de gewichten $w_k(y^q)$ en de schattingen $\hat{p}_k(x|y^k)$.

Voor een gegeven voorgeschiedenis y is er een maximum $r = r_y \leq q$ voor de lengte van de overeenkomst tussen de context y^r en een tak van de trie. Omdat er voor hogere ordes geen adequate probabiliteitsschattingen zijn, is $w_k(y^q) = 0$ voor $k > r$ en is $w_k(y^q) = w_k(y^r)$. Voor $k \leq r$ is de context y^k zeker al voorgekomen, maar is het niet noodzakelijk zo dat ook de combinatie $y^k x$ al opgetreden is. Wordt de relatieve conditionele frequentie gebruikt als probabiliteitsschatting, dan zou ook hier het nulfrequentieprobleem kunnen optreden indien $\hat{p}_k(x|y^k) = 0$ voor alle k. Maar de aanwezigheid van het orde-(-1) model garandeert dat er altijd minstens één model een bruikbare probabiliteitsschatting oplevert. Indien $\hat{p}_k(x|y^k) \neq 0$, dan geeft het orde-k model een *geldige voorspelling* voor x gegeven de context y^k. De context y^k is een *geldige context* indien er minstens één waarde x bestaat waarvoor $\hat{p}_k(x|y^k) \neq 0$.

Figuur 4.2 schetst een kleine trie van maximale orde $q = 3$ voor een bronverzameling $\mathcal{X} = \{a, b, c, d\}$. Is y = '*aadb*' en x ='*c*', dan is $r = 2$ en is y^r = '*db*' een geldige context. De context bij elk kind bestaat uit de combinatie van de context van de ouder en een nieuw karakter. Bij elke knoop hoort een frequentietabel.



Figuur 4.2: Een trie van maximale orde $q = 3$. De notatie $p_k(X|y^k)$ is equivalent aan $\{\hat{p}_k(x|y^k);\ x \in \mathcal{X}\}$ en stelt de frequentietabel voor die hoort bij de context y^k.

Escapemechanisme

Wat betreft de gewichten $w_k(y^r)$ en de probabiliteitsschattingen $\hat{p}_k(x|y^k)$ zijn veel gemotiveerde keuzes mogelijk. Zonder verregaande veronderstellingen te maken is geen enkele keuze theoretisch te verkiezen, maar toch zijn er enkele vuistregels die intuïtief duidelijk zijn. Een goede keuze voor de probabiliteitsschattingen is de relatieve frequentie van optreden. Langere contexten hebben een grotere voorspellende waarde, zodat $w_k(y^r)$ bij voorkeur groot is voor grote k indien de context y^k veel voorgekomen is of indien weinig verschillende waarden x erop volgen.

Het is dan ook logisch om een onderscheid te maken op basis van het feit of x in het verleden al dan niet is opgetreden na een gegeven context y^k. Alle nog niet opgetreden bronwaarden bij een context van orde k worden gegroepeerd als het *escapesymbool* $e_{|y^k}$ of kortweg e. De waarschijnlijkheid van deze artificiële bronwaarde, de *escapewaarschijnlijkheid* $\hat{p}(e|y^k)$, wordt intuïtief gekozen en drukt de waarschijnlijkheid uit dat het contextmodel van orde k niet in staat is een behoorlijke schatting te genereren.

Stelt $c(y^k x)$ het aantal keren voor dat de context y^k gevolgd werd door de

waarde x, dan is de context y^k al $\bar{c}(y^k) = \sum_{x \in \mathcal{X}} c(y^k x)$ keer voorgekomen. Verder bevat $\mathcal{X}_{|y^k}$ alle waarden x van $\mathcal{X}$ waarvoor $y^k x$ voorgekomen is en wordt het complement $\bar{\mathcal{X}}_{|y^k} = \mathcal{X} \setminus \mathcal{X}_{|y^k}$ afgebeeld op het escapesymbool e voor een context y^k van lengte k. Tenslotte bevat $\mathcal{X}^j_{|y^k}$ alle waarden x van $\mathcal{X}$ die precies j keer de context y^k volgden.

De gewichten moeten voldoen aan $\sum_{k=-1}^{r} w_k(y^r) = 1$. Gebaseerd op het escapemechanisme is hieraan voldaan indien de gewichten gekozen worden als

$$\begin{aligned} w_r(y^r) &= 1 - \hat{p}(e|y^r) \\ w_k(y^r) &= (1 - \hat{p}(e|y^k)) \prod_{t=k+1}^{r} \hat{p}(e|y^t) \qquad \text{voor } -1 \leq k < r. \end{aligned}$$

De factor $(1 - \hat{p}(e|y^k))$ drukt uit in welke mate het model van orde k in staat is een relevante voorspelling te maken, terwijl de normeringsfactor $\prod_{t=k+1}^{r} \hat{p}(e|y^t)$ het gewicht doet afnemen indien modellen van hogere orde eveneens een relevante voorspelling maken. Bij conventie is $\hat{p}(e|y^{-1}) = 0$, wat betekent dat modellen van orde $k < -1$ niet in rekening gebracht worden en dat een model van orde -1 altijd een geldige voorspelling maakt.

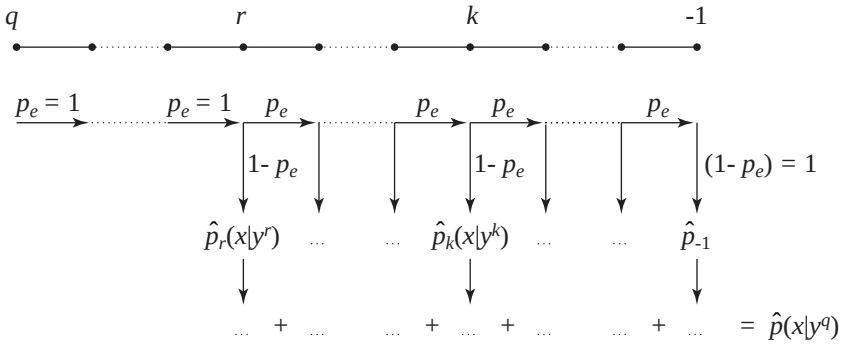
Deze keuze van de gewichten $w_k(y^r)$ kan nu terug ingevoerd worden in de oorspronkelijke definitie voor de gemengde probabiliteitsschatting $\hat{p}(x|y^q)$, wat resulteert in

$$\hat{p}(x|y^q) = \sum_{k=-1}^{r} \prod_{t=k+1}^{r} \hat{p}(e|y^t)\,(1 - \hat{p}(e|y^k))\,\hat{p}_k(x|y^k).$$

Deze uitdrukking beschrijft hoe het "mengen" van de geschatte probabiliteiten precies verloopt. Elk model van orde k, waarbij k varieert van r tot -1, genereert een bijdrage tot de gemengde probabiliteit $\hat{p}(x|y^q)$ gegeven door

$$w_k(y^r)\,\hat{p}_k(x|y^k) = \left[\prod_{t=k+1}^{r} \hat{p}(e|y^t) \right] \cdot \left[(1 - \hat{p}(e|y^k))\right] \cdot \left[\hat{p}_k(x|y^k)\right].$$

De bijdrage bestaat dus uit het product van drie factoren "$[\cdot]$", die respectievelijk geïnterpreteerd kunnen worden als de probabiliteit om te dalen ("ontsnappen") van het orde-r tot het orde-k model, *en* de probabiliteit om niet verder te dalen, *en* de probabiliteit om de bronwaarde x te selecteren binnen het model van orde k. Figuur 4.3 illustreert hoe de modellen van verschillende orde via het escapemechanisme elk hun gewogen bijdrage leveren tot het gemengde model.



Figuur 4.3: Voor een gegeven context y worden de probabiliteitsschattingen $\hat{p}_k(x|y^k)$ horend bij modellen van verschillende orde k gewogen en samengesteld tot een gemengde schatting $\hat{p}(x|y^q)$. Het escapemechanisme bepaalt de geschatte probabiliteit $p_e = \hat{p}(e|y^k)$, die het onvermogen uitdrukt van het model van orde k om een relevante voorspelling te maken.

De *coderuimte* (*E.* code space) toegekend aan x in het orde-k model is $(1-\hat{p}(e|y^k))\ \hat{p}_k(x|y^k)$. De coderuimte kan beschouwd worden als de geschatte probabiliteit voor $x \in \mathcal{X}_{|y^k}$ genormeerd over $\mathcal{X}_{|y^k} \cup \{e_{|y^k}\}$ terwijl $\hat{p}_k(x|y^k)$ de geschatte probabiliteit voorstelt genormeerd over $\mathcal{X}$. Een eenvoudige uitdrukking voor de coderuimte is belangrijk voor de snelheid omdat deze telkens opnieuw berekend wordt. De keuze van de escapewaarschijnlijkheid $\hat{p}(e|y^k)$ heeft op deze manier een grote impact op de bekomen snelheid en efficiëntie.

Het escapemechanisme laat toe de gewichten op een indirecte en intuïtief aanvaardbare manier toe te kennen. De modellen die voortvloeien uit het gebruik van het escapemechanisme passen binnen het formalisme van gemengde modellen, maar kunnen ook anders geïnterpreteerd worden. Bovendien zijn ze veel eenvoudiger te implementeren in tegenstelling tot de gemengde modellen in hun meest algemene vorm, die veel geheugen en rekenwerk vereisen.

Wat de probabiliteiten betreft, moet zowel een schatting $\hat{p}_k(x|y^k)$ voor $x \in \mathcal{X}_{|y^k}$ als een schatting $\hat{p}(e|y^k)$ gemaakt worden. De keuze voor $\hat{p}(e|y^k)$ is niet eenduidig en bepaalt de gewichten $w_k(y^r)$ via het escapemechanisme zoals hierboven beschreven. De schattingen $\hat{p}_k(x|y^k)$ zijn genormeerd over $\mathcal{X}_{|y^k}$. Voor deze keuzes zijn enkele alternatieven voorgesteld in de literatuur [14, 240]. Hierna is $l_k = l(\mathcal{X}_{|y^k})$ en $l_k^{(j)} = l(\mathcal{X}^j_{|y^k})$, stelt l_0 het aantal elementen in $\mathcal{X}$ voor dat in het verleden al minstens één keer opgetreden is en is per definitie $l_{-1} = l(\mathcal{X})$.

- **methode A**, zie [39, 274]: Het escapesymbool wordt beschouwd als eenmalig opgetreden en de relatieve frequentie wordt gebruikt als pro-

babiliteitsschatting, zodat $\hat{p}(e|y^k) = 1/(\bar{c}(y^k)+1)$ en $\hat{p}_k(x|y^k) = c(y^kx)/\bar{c}(y^k)$. De coderuimte voor x in het orde-k model vereenvoudigt tot $c(y^kx)/(\bar{c}(y^k)+1)$.

- **methode B**, zie [39, 274]: Het escapesymbool krijgt een telwaarde toegewezen die gelijk is aan het aantal verschillende waarden x volgend op een context y^k, zodat $\hat{p}(e|y^k) = l_k/\bar{c}(y^k)$. In tegenstelling tot de vorige methode wordt nu dus rekening gehouden met de veranderlijkheid van de waargenomen opvolgende waarden x. Deze aanpak heeft zin indien l_k klein is ten opzichte van $l(\mathcal{X})$. Indien l_k in de buurt komt van $l(\mathcal{X})$ wordt een tegengesteld gedrag verwacht omdat de probabiliteit van het escapesymbool dan afneemt naarmate er meer waarden waargenomen zijn. Bovendien wordt het eerste optreden van een waarde x niet in rekening gebracht voor de probabiliteitsschatting zodat 1 afgetrokken wordt van de telwaarde. Zonder het escapesymbool in rekening te brengen wordt de probabiliteit $\hat{p}_k(x|y^k) = (c(y^kx)-1)/(\bar{c}(y^k)-l_k)$. Dit betekent dat uitzonderlijke, eenmalige gebeurtenissen niet in rekening gebracht worden. De coderuimte voor x in het orde-k model neemt ook hier een eenvoudige gedaante aan, namelijk $(c(y^kx)-1)/\bar{c}(y^k)$.

- **methode C**, zie [158]: Deze methode is gelijkaardig aan methode B, maar het eerste optreden van een bronwaarde x wordt nu wel in rekening gebracht. Er volgt dat $\hat{p}(e|y^k) = l_k/(\bar{c}(y^k)+l_k)$ en $\hat{p}_k(x|y^k) = c(y^kx)/\bar{c}(y^k)$. De coderuimte voor x in het orde-k model is $c(y^kx)/(\bar{c}(y^k)+l_k)$.

- **methode D**, zie [97]: Een eerste optreden van y^kx wordt voor de helft toegeschreven aan de bronwaarde x en voor de andere helft aan het escapesymbool e. Dit geeft aanleiding tot $\hat{p}(e|y^k) = l_k/(2\,\bar{c}(y^k))$ en $\hat{p}_k(x|y^k) = (c(y^kx)-1/2)/\bar{c}(y^k)$.

- **methode P, X, XC en X1**, zie [274]: Methode P vertrekt van de veronderstelling dat de escapesymbolen gegenereerd zijn door een poissonproces. Hieruit volgt $\hat{p}(e|y^k) = l_k^{(1)}/\bar{c}(y^k) - l_k^{(2)}/\bar{c}^2(y^k) + l_k^3/\bar{c}^{(3)}(y^k) - \cdots$. Merk op dat deze aanpak negatieve schattingen kan opleveren, zodat de methode nog een aantal verfijningen nodig heeft. Methode X beperkt voorgaande som tot de eerste term. De probabiliteiten moeten genormeerd worden en ook hier zijn speciale maatregelen nodig, meer specifiek voor het geval $l_k^{(1)} = 0$ of $l_k^{(1)} = \bar{c}(y^k)$ [163]. De methode XC gebruikt methode X indien mogelijk en valt in het andere geval terug op methode C. Methode X1 tenslotte omzeilt de problemen van methode X

door de telwaarden te verhogen, zodat $\hat{p}(e|y^k) = (l_k^{(1)} + 1)/(\bar{c}(y^k) + l_k^{(1)} + 1)$. De coderuimte tenslotte wordt $c(y^k x)/(\bar{c}(y^k) + l_k^{(1)} + 1)$ [240].

Uitsluiting en herschaling

Volledig gemengde modellen, waarbij iedere orde k een bijdrage heeft in de geschatte probabiliteit, vragen zeer veel rekenkracht. Bovendien zijn cumulatieve schattingen nodig voor de entropiecodering en moeten de berekeningen met hoge precisie gebeuren omwille van de soms zeer kleine waarden voor de probabiliteitsschatting. Deze problemen kunnen vermeden worden door gebruik te maken van *volledige uitsluiting* (*E.* full exclusion) voor de voorspelling. Enkel het model van hoogste orde r^* dat een geldige voorspelling geeft voor x wordt in rekening gebracht. De modellen van lagere orde $k < r^*$ geven ook een geldige voorspelling voor x, maar worden "uitgesloten" voor de probabiliteitsschatting. Merk op dat er modellen kunnen bestaan van nog hogere orde dan r^* die een geldige voorspelling maken, zij het voor andere waarden dan x. De waarden z waarvoor een geldige voorspelling kan gemaakt worden bij nog hogere ordes $k > r^*$ worden niet in rekening gebracht voor alle tussenliggende ordes inclusief orde r^*. Het uitsluitingsprincipe wordt gecombineerd met een escapemethode uit vorige paragraaf.

Volledige uitsluiting gaat als volgt tewerk. Stel dat een context y^q en een waarde x gezamenlijk optreden en dat r de langste overeenkomst is tussen de context en het gemengd model. Dan is initieel $k = r$ en herhaalt de volgende cyclus zich. Is $y^k x$ nog niet opgetreden, dan wordt het escapesymbool $e_{|y^k}$ doorgestuurd met probabiliteit $\hat{p}(e|y^k)$, wordt elk optreden van $z \in \mathcal{X}_{|y^k}$ niet langer in rekening gebracht voor alle modellen van lagere orde en herbegint de cyclus met een model van orde $k-1$. Het niet in rekening brengen van een waarde z voor alle modellen van orde $t < k$ wordt bekomen door (tijdelijk) $c(y^t z) = 0$ te stellen voor $t < k$. Is daarentegen $y^k x$ wel al opgetreden voor een bepaalde k, dan is per definitie $r^* = k$ en wordt de coderuimte $(1 - \hat{p}(e|y^k))\,\hat{p}_k(x|y^r)$ gebruikt als gecorrigeerde probabiliteitsschatting voor x. De cyclus stopt en alle schattingen voor ordes $k < r^*$ worden uit het model uitgesloten. De aanwezigheid van het orde-(-1) model garandeert dat de cyclus stopt.

Uitsluiting heeft als nadeel dat statistische bemonsteringsfouten versterkt worden. Wordt methode A, B of C gebruikt dan zal bovendien, indien alle waarden van $\mathcal{X}$ voorgekomen zijn bij een model van orde k, toch nog $\hat{p}(e|y^k) \neq 0$ en dit is een verspilling van coderuimte. Het verlies is klein voor grote bronverzamelingen maar kan belangrijk worden voor kleine bronverzamelingen. Anderzijds zijn de algoritmen veel sneller en is de implementatie

aanzienlijk vereenvoudigd.

Een verdere vereenvoudiging van gemengde modellen wordt bekomen met *luie uitsluiting* (*E.* lazy exclusion) voor de voorspelling. Net als bij uitsluiting wordt enkel het model van hoogste orde r^* gebruikt dat een geldige voorspelling geeft voor x en laat het escapemechanisme toe om de hogere ordes te coderen. Maar het aandeel van de waarden z waarvoor een geldige voorspelling bestaat bij een nog hogere orde $k > r^*$, wordt wel in rekening gebracht. Dit is een verspilling van coderuimte omdat deze waarden niet kunnen optreden. De efficiëntie vermindert maar de snelheid neemt toe.

Bij het updaten van de frequentietabellen wordt het optreden $y^q x$ geteld in alle modellen van orde $0 \leq k \leq r$. Uitsluiting gebruikt echter enkel het model van hoogste orde dat een geldige voorspelling voor x maakt. Het kan voordelig zijn om enkel de frequentietabel van die orde aan te passen en de andere ongewijzigd te laten. Als probabiliteitsschatting wordt dan niet de zuivere relatieve frequentie gebruikt, maar enkel de relatieve frequentie van de waarden die niet voorspeld worden door modellen van hogere orde. Dit wordt *update-uitsluiting* (*E.* update exclusion) genoemd en zorgt voor een kleine compressieverbetering.

De meeste sequenties die men in de praktijk tegenkomt voldoen niet aan de voorwaarde van stationariteit. In tekst bijvoorbeeld treedt er een zekere mate van "recentelijkheid" op, dit wil zeggen dat recent opgetreden woorden een verhoogde kans tonen om in de nabije toekomst opnieuw op te treden, wat op zijn beurt een impact heeft op de statistieken van het karaktergebaseerde contextmodel [131]. Qua uitgestrektheid overtreft deze recentelijkheid de maximale orde van het model zodat het niet als een markoveigenschap gemodelleerd kan worden. Maar anderzijds is deze uitgestrektheid veel kleiner dan de volledige lengte van de bronsequentie zodat ze als een uiting van niet-stationariteit moet geïnterpreteerd worden. Een extreem voorbeeld van deze niet-stationariteit is bijvoorbeeld het gebruik van verschillende talen binnen eenzelfde document. Een *herschaling* van het model verkleint alle telwaarden op een evenredige manier, speciale voorzieningen om het nulfrequentieprobleem te vermijden niet te na gesproken. Dit zorgt ervoor dat het relatief belang van de oudere gebeurtenissen in de statistieken verkleint en dat het bekomen model zich vlugger aanpast aan de gewijzigde statistieken. Het effect van herschaling is equivalent aan het concept van *verouderen* (*E.* aging), waarbij het relatief belang van de gegevens afneemt volgens hun ouderdom.

Implementaties

Volledige vermenging van de modellen komt in praktische implementaties niet voor. Het escapemechanisme laat immers toe deze bewerkelijke stappen te omzeilen, al blijven de resulterende implementaties voldoen aan de eigenschappen van een gemengd model. Bij een praktische implementatie zijn naast de verwerkingssnelheid ook de benodigde hoeveelheid geheugen belangrijk.

Op basis van de hierboven besproken methoden en in combinatie met een maximale orde q en eventuele uitsluiting en herschaling worden praktische compressiealgoritmen opgesteld. De meeste zijn gebaseerd op Prediction by Partial Matching of PPM. Zo combineren PPMA en PPMB respectievelijk methode A en B met volledige uitsluiting, maar gebruiken ze geen herschaling. De bekendste PPM-techniek is PPMC, die gebaseerd is op het escapemechanisme van methode C en die verder update-uitsluiting gebruikt en de telwaarden herschaalt tot een maximale precisie van 8 bit [159]. PPMC' is een qua snelheid geoptimaliseerde variant van PPMC die luie uitsluiting gebruikt voor de voorspelling, update-uitsluiting en bovendien een bovengrens oplegt aan het geheugengebruik. PPMD is gebaseerd op methode D en maakt eveneens gebruik van uitsluiting, maar niet van herschaling [97]. Op tekstbestanden presteren PPMC en PPMD keer op keer een stuk beter dan PPMA en PPMB. Het schalen van de telwaarden gebaseerd op hun ouderdom, het schalen van de telwaarden bij deterministische contexten en het invoeren van twee contextgestuurde probabiliteitsvoorspellers om niet-redundante data te detecteren, resulteert in PPMD+ [239]. Deze scoort consequent als beste van de tot hiertoe besproken PPM-technieken. Tot slot is er de vrij snelle implementatie PPMC-h. Ze is gebaseerd op PPMC, maar maakt gebruik van hashtabellen met gelinkte lijsten in plaats van tries, een bereikcoder, update-uitsluiting en luie of volledige uitsluiting [28].

Alle PPM-technieken hebben één parameter gemeen die vooraf vastgelegd moet worden: de maximale orde q van het gemengd model. Voor tekst situeert het optimum zich in de praktijk rond $q = 5$ en de efficiëntie neemt vlug af bij hogere waarden. Een minder evidente parameter voor tekstcompressie is de duidelijke aflijning van het alfabet.

Varianten

Gelijkaardige ideeën werden eerder al theoretisch beschreven en specifiek uitgewerkt voor binaire beelden in het algoritme *Context* [191]. Voor een ergodisch markovproces wordt er een *sorteerfunctie* (*E.* sorting function) voorgesteld die de indices van de conditionerende toevalsgrootheden permuteert en op deze manier het verleden herordent. Deze sorteerfunctie laat toe om

flexibel met het begrip nabuurschap om te gaan indien toevalsgrootheden gemodelleerd worden waarvan de index van een hogere dimensie is. Concreet zorgt de sorteerfunctie er bijvoorbeeld bij beeldcompressie voor dat de pixels links en boven de huidige pixel als "nabij" en de andere als "veraf" gedefinieerd kunnen worden. Een *sequentie* (*E.* string) $s = x_i^i = x_1 x_2 \cdots x_i$ stelt er de aaneenrijging voor van bronwaarden van de beginindex 1 tot en met de huidige index i. De *structuurfunctie* (*E.* structure function) beeldt elke eindige sequentie af op een natuurlijk getal. Het begrip *context* van een bronwaarde x_i wordt er formeel gedefinieerd als de equivalentieklasse van alle sequenties die door de structuurfunctie op hetzelfde getal afgebeeld worden als de "voorbije" sequentie x_{i-1}^{i-1}. De probabiliteiten van verschillende orde worden niet gemengd, maar één specifieke orde wordt geselecteerd aan de hand van een criterium dat gericht is op een minimale conditionele entropie.

De techniek *Dynamic History Predictive Compression* of DHPC is een eindigecontextmodel dat principieel gelijkaardig is aan PPM maar zich onderscheidt op alle aspecten van de implementatie [271, 272]. Gedurende de verwerking van de sequentie wordt een achterwaartse contextboom opgebouwd die begrensd is door enkele vooraf vastgelegde limieten. Aan iedere knoop en blad wordt een context en een statistiek toegekend. De statistiek van een knoop is *geloofwaardig* (*E.* credible) indien zijn absolute frequentie de vooraf gedefinieerde *geloofwaardigheidsdrempel* (*E.* credibility threshold) overtreft. De boom kan groeien doordat een blad vervangen wordt door een knoop waaraan een nieuw aantal bladeren gekoppeld is. Een blad is *uitbreidbaar* (*E.* extensible) indien zijn absolute frequentie de *uitbreidbaarheidsdrempel* (*E.* extensibility threshold) overtreft. DHPC bouwt de contextboom trager op dan PPM en voorspelt de probabiliteit op basis van één enkele knoop. In uitvoering is het sneller maar minder efficiënt.

Ook *Dynamic Markov Coding* of DMC toont veel gelijkenis met PPM-technieken [40, 96]. De bronsequentie wordt gemodelleerd als een *eindige automaat* (*E.* finite-state machine) of FSM. Deze wordt gekenmerkt door een eindige verzameling toestanden $\mathcal{S} = \{s_i\}$, een starttoestand s^*, een bronalfabet $\mathcal{X}$ en een toestandstransitiefunctie $f : \mathcal{S} \times \mathcal{X} \mapsto \mathcal{S}$. De conditionele probabiliteiten voor de uitkomsten $\{p(x_i|s_j)\}$ worden geschat en gebruikt om een aritmetische coder aan te drijven. DMC start van een eenvoudig en generisch initieel model met weinig toestanden. Vervolgens voegt het algoritme toestanden toe gedurende het verwerken van de sequentie. Dit gebeurt door een bestaande toestand, waarvoor voldoende variatie is in de voorgaande toestanden, te *klonen*. Parallel met het beheren van de toestanden staat DMC in voor het schatten van de probabiliteiten van de mogelijke uitkomsten voor elke toestand.

Elk tot hiertoe besproken eindigecontextmodel (FCM) is ook een eindige automaat (FSM). Maar omdat FSM's potentieel een oneindig geheugen kunnen hebben, is niet elke FSM ook een FCM. Sequenties waarbij de waarden afhangen van hun index, ook *telgebeurtenissen* (*E.* counting events) genoemd, zijn hier een mooi voorbeeld van. Zo kan de binaire sequentie waarvan elke even waarde '1' is en waarvan elke oneven waarde arbitrair '0' of '1' kan zijn, nooit perfect gemodelleerd worden als een FCM, maar als FSM is het triviaal. Toch zorgt het proces van klonen ervoor dat DMC enkel FSM's genereert die ook FCM's zijn [13]. Op deze manier blijft de theoretisch haalbare efficiëntie van DMC begrensd door de klasse van de FCM's, waartoe ook de PPM-modellen behoren. Technieken gebaseerd op DMC gebruiken in theorie meer geheugen dan PPM-gebaseerde technieken maar ze zijn iets sneller.

Uitbreidingen

De maximale lengte van een context in een eindigecontextmodel is een parameter die een groot belang heeft op de bekomen compressie. De beste resultaten voor tekstcompressie worden bekomen voor een maximum van $q = 5$ karakters. PPM* is een uitbreiding van PPM die geen beperkingen oplegt aan de contextlengte [38]. Een context is *deterministisch* als die precies één geldige voorspelling doet. De methode selecteert de kortste deterministische context. Indien er geen deterministische context bestaat, wordt de langste geldige context gebruikt. Deze selectie van de contextorde wordt gecombineerd met een escapemechanisme. De geheugenvereisten zijn veel hoger dan die van klassieke PPM-technieken. Om deze te beperken wordt de context-trie omgezet in een *patricia-trie* (*E.* practical algorithm to retrieve information coded in alphanumeric). Hierbij worden alle paden die zich niet verder vertakken vervangen door één enkel blad.

PPMZ vormt een verdere uitbreiding van deze techniek [17] en is gebaseerd op heuristieken. Ten opzichte van PPM of PPM* zijn er drie wijzigingen voorgesteld. Ten eerste moeten deterministische contexten van onbegrensde lengte voldoende lang zijn om in aanmerking te komen. Ten tweede hoort bij elke orde een context en een meest waarschijnlijk symbool (*E.* most probable symbol) of MPS. De orde waarvoor de waarschijnlijkheid van dit MPS maximaal is wordt gekozen als lokale orde. Ten derde wordt de schatting van de waarschijnlijkheid van het escapesymbool niet rechtstreeks gebruikt, maar wordt deze schatting opgenomen in de context. Geconditioneerd op deze context worden vervolgens correcte statistieken voor het escapesymbool opgebouwd. Deze aanpassingen geven aanleiding tot een hele klasse van algoritmen. Tenslotte merken we op dat PPMZ2 nog een aantal verdere verfijningen

toevoegt aan deze aanpak.

De aanpak van *Context Tree Weighting* of CTW is bijzonder efficiënt en elegant voor binaire sequenties. De methode maakt geen veronderstellingen voor het bronmodel behalve dat het net als PPM een bovengrens q oplegt voor de maximale contextlengte [269]. De binaire sequentie s wordt beschouwd als een aggregaat van subsequenties behorend bij verschillende subbronnen, elk corresponderend met een specifieke context y^k. Elke geconditionieerde subsequentie $s_{|y^k}$ bevat de waargenomen bits die corresporenderen met de respectieve waarnemingen van de context y^k. Op deze manier hoort bij elke context y^k een aantal waargenomen '0'-bits en '1'-bits, respectievelijk als $a(y^k)$ en $b(y^k)$ genoteerd. De contexten y^k worden georganiseerd in een *contextboom* (*E.* context tree) van een vooropgestelde maximale diepte. Bij elk blad en elk knooppunt van deze contextboom hoort een context y^k en dus ook een subsequentie $s_{|y^k}$ waaruit $a(y^k)$ en $b(y^k)$ kunnen bepaald worden. Deze worden gebruikt om een waarschijnlijkheid $p(s_{|y^k})$ te schatten voor de waargenomen subsequentie. Meer concreet wordt de Krichevski-Trofimov-schatter

$$\hat{p}_e(s_{|y^k}) = \frac{(a-1/2)(a-3/2)\cdots 1/2\,(b-1/2)(b-3/2)\cdots 1/2}{(a+b)(a+b-1)\cdots 1}$$

gebruikt omdat deze ideaal is voor een geheugenloze bron met ongekende parameter. Deze schattingen worden gecombineerd over de verschillende dieptes zodat gewogen probabiliteitsschattingen $\hat{p}_w$ worden bekomen:

$$\hat{p}_w(s_{|y^k}) = \begin{cases} \frac{1}{2}\hat{p}_e(s_{|y^k}) + \frac{1}{2}\hat{p}_w(s_{|0y^k})\hat{p}_w(s_{|1y^k}) & \text{als } l(y^k) < q, \\ \hat{p}_e(s_{|y^k}) & \text{als } l(y^k) = q. \end{cases}$$

Er kan eenvoudig aangetoond worden dat de extra kost voor het mengen van twee schattingen volgend uit verschillende modellen ten hoogste 1 bit bedraagt. Voorgaande uitdrukking wordt recursief toegepast op de contextboom om de gewogen probabiliteit $\hat{p}_w(s_{|\Lambda})$ van de hele sequentie te bepalen, waarbij Λ de sequentie met lengte 0 voorstelt en dus de wortel van de boom definieert. Het coderen van deze globale waarschijnlijkheid $\hat{p}_w(s_{|\Lambda})$ gebeurt via zijn aritmetische code met een extra kost van twee bits. In een praktisch compressieschema moet ook het model gespecificeerd worden inclusief de parameters en de statistieken, wat nog een belangrijke extra kost betekent. De methode werd later uitgebreid naar andere bronklassen [270]. Omdat deze methode een probabiliteit schat voor de hele sequentie is ze strikt genomen niet karaktergebaseerd maar blokgebaseerd, waarbij de hele sequentie als één lang blok beschouwd wordt. De methode is geniaal in haar eenvoud, maar schiet tekort op het vlak van implementatiemogelijkheden.

Symboolsortering (*E.* symbol ranking) gebruikt eveneens contexten van veranderlijke lengte maar schat de waarschijnlijkheden niet expliciet en maakt

geen gebruik van escapesymbolen [75]. De elementen van $\mathcal{X}_{|y^k}$ worden impliciet gesorteerd volgens afnemende geschatte waarschijnlijkheid en de index van de waargenomen waarde x wordt gecodeerd. Eerst wordt de langste geldige context geprobeerd en indien die geen geldige voorspelling heeft voor x wordt de lengte van de context systematisch verlaagd tot een context bereikt wordt die een geldige voorspelling maakt voor x.

De hierboven beschreven PPM-varianten schatten de waarschijnlijkheid van het escapesymbool op een eerder arbitraire manier. Het is mogelijk dit op een meer gefundeerde manier te doen door een parametrische vorm voor de escapewaarschijnlijkheid voorop te stellen [1]. Vervolgens wordt aan de parameters een waarde toegekend door een compactheidscriterium toe te passen op een aantal representatieve sequenties. Bijkomend voordeel is dat via deze parameters de schatting van de escapewaarschijnlijkheid adaptief kan wijzigen tijdens de sequentie.

De toekenning van de gewichten aan de contextmodellen van verschillende orde kan verder gegeneraliseerd en verbeterd worden [22]. PPM-modellen proberen meestal eerst de hoogste ordes die een geldige voorspelling maken, maar dat is doorgaans niet de beste keuze. Met elk knooppunt van een contextboom kan een *toestand* (*E.* state) geassocieerd worden, en als huidige context kan gekozen worden voor de toestand die in het verleden een minimaal bericht genereerde. Een mogelijke manier om deze beste toestand te zoeken is via dynamisch programmeren, waarbij het vermogen van een kindknooppunt *doorsijpelt* naar het ouderknooppunt [23]. Tot slot valt op te merken dat een taxonomie van de in de literatuur voorgestelde contextgebaseerde compressietechnieken aantoont dat ze in wezen gelijkaardige bouwstenen gebruiken en binnen één kader kunnen gestructureerd worden [21].

Wisselschema's

Een sequentieel universeel broncoderingsalgoritmen zoals PPM schat de waarschijnlijkheid voor elk karakter van de sequentie. Indien er meerdere algoritmen voorhanden zijn, kunnen deze schattingen gecombineerd worden. De sequentie wordt opgesplitst in blokken van veranderlijke lengte en voor elk blok wordt de methode gebruikt die hierop de hoogste efficiëntie haalt. Indien deze opsplitsing causaal is, zal echter weinig of geen winst in compressie bekomen worden. Indien deze opsplitsing niet causaal is, moet ze expliciet als zij-informatie naar de decoder doorgestuurd worden. De lengte van de entropiegecodeerde stroom wordt dan wel geminimaliseerd, maar deze winst kan tenietgedaan worden door de extra zij-informatie. Daarom wordt de opsplitsing in blokken en toekenning van een methode globaal geoptimaliseerd.

De resulterende methoden worden *wisselschema's* (*E.* switching schemes) genoemd, VW98 vormt hiervan het bekendste voorbeeld [261]. Ze bevinden zich op de brug tussen karaktergebaseerde en blokgebaseerde methoden.

4.2.2 Blokgebaseerde methoden

De *blokgebaseerde* methoden verwerken een blok karakters tegelijkertijd. Een *blok* bestaat uit één of meerdere karakters. Woordenboekgebaseerde methoden gebruiken blokken van veranderlijke lengte, terwijl bloksorteringsmethoden blokken van vaste lengte verwerken. Blokgebaseerde methoden zijn ouder en meestal sneller dan karaktergebaseerde en vormen daarom de basis van de populaire praktische compressieprogramma's voor algemeen gebruik zoals "compress", "gzip", "pkzip" en "winzip".

Woordenboekgebaseerde methoden

Woordenboekgebaseerde compressiealgoritmen verwerken de sequentie in blokken. Een *woordenboek* (*E.* dictionary) van veel voorkomende patronen wordt geïnitialiseerd, opgebouwd en onderhouden. De blokgrenzen worden bepaald door de langste *overeenkomst* (*E.* match) tussen de toekomstige sequentie en het woordenboek. De overeenkomst wordt gecodeerd aan de hand van een *verwijzing* (*E.* pointer) naar het woordenboek. Typisch is dit de index van het overeenkomende woord in het woordenboek. Het woordenboek kan statisch of adaptief zijn. Een *statisch* (*E.* static) woordenboek blijft ongewijzigd gedurende het verwerken van de sequentie. Een *adaptief* (*E.* adaptive) woordenboek verandert doorgaans tussen elke twee blokken. Het woordenboek kan expliciet of impliciet zijn. Een *expliciet* woordenboek betekent dat effectief een lijst met patronen opgebouwd en onderhouden wordt. Een *impliciet* woordenboek daarentegen bouwt de nodige informatie op wanneer het gewenst is. Dit kan bijvoorbeeld het reeds verwerkte gedeelte zijn van de sequentie.

LZ77 is een door Ziv en Lempel voorgestelde woordenboekgebaseerde methode met adaptief en impliciet woordenboek [288]. De methode maakt gebruik van een *schuivend venster* (*E.* sliding window) dat bestaat uit een *zoekbuffer* (*E.* search buffer) die een recent stuk van het verleden bevat en een *voorwaarts buffer* (*E.* look-ahead buffer) die de toekomstige evolutie voorstelt. Er wordt gezocht naar de langste overeenkomst tussen enerzijds de voorwaartse buffer en anderzijds alle subsequenties van de zoekbuffer. De overeenkomst wordt gecodeerd als een triplet bestaande uit: de positie p van de overeenkomst binnen de zoekbuffer, de lengte l van de overeenkomst en het eerste verschillend karakter c binnen de voorwaartse buffer. Dit laatste is nodig om uit

de situatie te kunnen ontsnappen waarbij geen overeenkomst gevonden wordt. Daarna verschuift het venster $l + 1$ posities verder. Het decoderen verloopt eenvoudiger en sneller omdat het zoeken naar de langste overeenkomst vervalt. Om die reden is LZ77 een asymmetrische methode. De LZ77-techniek ligt aan de grondslag van het populaire GZIP.

Een nieuw karakter wordt inefficiënt gecodeerd omdat er drie parameters doorgestuurd worden terwijl er slechts één van belang is. Dit wordt opgelost in LZSS door een *vlagbit* (*E*. flag bit) toe te voegen die aanduidt of het om een nieuw karakter dan wel om een overeenkomst gaat [235]. Bovendien volstaat het nu om het koppel bestaande uit positie en lengte door te sturen in plaats van een triplet.

De slechtst denkbare sequentie voor LZ77 is een schijnbaar toevallige periodische sequentie waarvan de periode langer is dan de lengte van de zoekbuffer. De sequentie wordt geëxpandeerd in plaats van gecomprimeerd. De variant LZ78 maakt gebruik van een expliciet woordenboek en heeft het nadeel van een eindig zoekvenster niet [289]. Er wordt gestart met een leeg woordenboek. Net als bij LZ77 wordt er gezocht naar de langste overeenkomst s tussen het woordenboek en de nog te coderen sequentie. De overeenkomst wordt gecodeerd als een koppel bestaande uit de index i van de overeenkomst s in het woordenboek en het eerstvolgend karakter c na deze overeenkomst. Het woordenboek wordt uitgebreid met het patroon sc bestaande uit de aaneenrijging van de overeenkomst s en het nieuwe karakter c. Deze cyclus herhaalt zich totdat de hele sequentie verwerkt is. Om te vermijden dat het woordenboek ongelimiteerd kan groeien, wordt in de praktijk de groei stilgelegd of wordt het woordenboek op een gepast tijdstip opgeruimd.

Het expliciet toevoegen van het nieuwe karakter c aan het gecodeerde koppel kan inefficiënt verlopen, denk bijvoorbeeld aan de sequentie "$aaa \cdots a$". LZW (*E*. Lempel, Ziv, Welch) omzeilt deze beperking door enkel de index van de overeenkomst s te coderen [268]. Het eerstvolgend nieuwe karakter c wordt niet langer expliciet doorgestuurd maar als start voor een nieuwe overeenkomst beschouwd. Daartoe moet het woordenboek geïnitialiseerd zijn met het volledige alfabet. Het patroon sc bestaande uit de overeenkomst en het eerstvolgend nieuwe karakter wordt onmiddellijk aan het woordenboek van de coder toegevoegd. Maar de decoder kent dit nieuwe patroon pas nadat de *volgende* overeenkomst doorgestuurd is. Indien het laatst toegevoegde patroon onmiddellijk opnieuw optreedt, lijkt er dan ook een probleem op te treden gedurende het decoderen. Er wordt immers verwezen naar een patroon dat op dat moment nog niet volledig gekend is door de decoder. Maar deze tegenstelling is slechts schijn. De vaststelling dat alle karakters van dit patroon behalve het laatste gekend zijn en dat het laatste karakter identiek is aan het eerste biedt

een oplossing voor dit probleem. De aanpak van LZW vormt de basis voor het programma “compress” dat aanwezig is op elk UNIX-gebaseerd besturingssysteem.

LZ-methoden zijn universeel en asymptotisch optimaal voor ergodische stationaire bronnen, maar de convergentie is zeer traag en in de praktijk wordt de limiet niet bereikt. Heel veel toepassingen maken gebruik van deze LZ-methoden. Bovendien zijn er in de literatuur veel varianten en verbeteringen voorgesteld. Eén ervan, LZP, combineert contexten met overeenkomsten van variabele lengte. De positie van de overeenkomst of de index van het woordenboek moet niet gecodeerd worden maar wordt door de context bepaald [16]. LZFG is een andere variant waarbij letterlijke blokken afgewisseld worden met kopieerverwijzingen naar de zoekbuffer [77]. De implementatie maakt gebruik van start-stap-stop-codes en kent veel varianten. Ze is verregaand verfijnd en geoptimaliseerd zodat ze bijna zo goed is als PPMC maar veel sneller.

Het geavanceerde *Associative Coder of Buyanovsky* of ACB is een recente nog niet gepubliceerde techniek die contexten combineert met overeenkomsten van veranderlijke lengte. Het woordenboek bevat alle mogelijke contexten van veranderlijke en onbegrensde lengte en de ermee corresponderende evoluties, gesorteerd volgens de contexten. De *associatieve lijst* horend bij een gegeven context bestaat uit de waargenomen evoluties die corresponderen met deze context. De huidige evolutie wordt gecodeerd als een triplet bestaande uit de index van de overeenkomst binnen de associatieve lijst, de lengte van de overeenkomst en het eerste verschillend karakter.

Methoden voor blokken van vaste lengte

Een andere klasse van algoritmen splitst de sequentie op in blokken van vaste lengte en verwerkt deze blokken onafhankelijk van elkaar. Zonder twijfel de bekendste methode is *bloksortering* (*E.* block sorting). Deze techniek is gebaseerd op de reversibele *Burrows-Wheeler-transformatie* of BWT. Daarnaast wordt gebruik gemaakt van verplaats-naar-voor-codering of MTF-codering, evenals van een huffmancode of aritmetische code [24]. Populaire implementaties gebaseerd op bloksortering zijn BZIP, BZIP2 en SZIP.

De BWT is een reversibele transformatie die een blok b van lengte n omzet in een ander blok $B(b)$ en een index i waarvoor geldt $1 \leq i \leq n$. Het nieuwe blok $B(b)$ is een permutatie van b en wordt op de volgende manier bekomen. Eerst wordt een $n \times n$ matrix M opgebouwd waarvan de rijen bestaan uit alle cyclische verschuivingen van b. De rijen in M zijn lexicografisch geordend. Het getransformeerde blok $B(b)$ is de laatste kolom van M. De index van de rij die correspondeert met het oorspronkelijk blok bepaalt i. Het is verre

van triviaal dat deze transformatie reversibel is, maar het bewijs wordt hier achterwege gelaten, zie [24]. Omdat de eerste kolom alfabetisch geordend is en omdat de rijen cyclisch geïnterpreteerd worden, komen de gelijke karakters in blok $B(b)$ in geconcentreerde vorm samen indien de tekstsequentie een mate van redundantie vertoont.

Bij *verplaats-naar-voor*-codering (*E.* move-to-front coding) of MTF-codering worden de karakters van $B(b) = y_1 y_2 \cdots y_n$ sequentieel verwerkt. Op ieder moment (of index) $i = 1..n$ worden de karakters gesorteerd volgens toenemend tijdsinterval sinds hun laatste optreden binnen het blok $B(b)$, zodat de meest recente karakters vooraan staan. Aan elk karakter y_i wordt vervolgens een index toegekend volgens deze gesorteerde lijst. Doordat gelijke karakters in $B(b)$ in geconcentreerde vorm voorkomen, zal de toegekende index dikwijls klein zijn. Daarom worden deze indices gecodeerd aan de hand van een huffman- of aritmetische code.

Bloksortering is van nature uit niet adaptief binnen een blok, maar wel over de blokken heen. De aanpak lijkt op het eerste zicht volkomen verschillend van die van PPM* maar toch kan aangetoond worden dat ze min of meer dezelfde manipulaties uitvoeren [38]. Ze zoeken namelijk allebei naar een kortste deterministische context voor elk karakter x_i. Bij PPM* zit die verborgen in de context-trie terwijl die bij bloksortering in de matrix M aanwezig is. Het enige verschil is dat PPM* de sequentie sequentieel en adaptief verwerkt daar waar bloksortering ieder karakter binnen een blok op dezelfde basis beschouwt.

4.3 Methoden voor binaire beelden

Compressiealgoritmen speciaal ontworpen voor beelden onderscheiden zich van algemene compressiealgoritmen doordat ze een veel betere interpretatie geven aan de beeldinformatie. Het tweedimensionaal karakter, de woordbreedte van de pixelintensiteit (1, 8, 12, 16, 24 of 36 bit), de hiermee gepaard gaande grootte van het bronalfabet, de aanwezigheid van ruis, de kwantitatieve interpretatie van de pixelintensiteit en de eventuele kleurinterpretatie worden expliciet in rekening gebracht. Zo falen de woordenboekgebaseerde technieken op de aanwezigheid van ruis en heeft de grootte van het bronalfabet een nefast effect op de convergentiesnelheid van contextgebaseerde methoden.

Binaire beelden gebruiken maar 1 bit per pixel en om die reden vormen ze interessante bronsequenties voor onderzoek. Een aantal problemen bij tekstcompressie, veroorzaakt doordat er 256 verschillende waarden kunnen optreden, worden op die manier vermeden. Eerst stellen we een nieuw verfijnd paradigma voor dat veralgemeend de compressie van bronsequenties beschrijft. Daarna komen achtereenvolgens looplengtecodering, contextmodellering, pa-

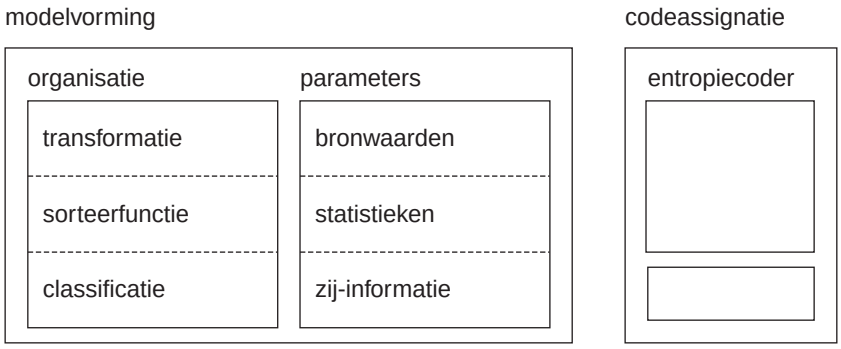
troongebaseerde en structurele technieken aan bod. Deze opsplitsing is evenwel niet eenduidig noch mutueel exclusief en veel methoden uit de praktijk combineren deze technieken dan ook.

4.3.1 Paradigma voor compressie

Omdat beelden als informatiestroom structureel verschillen van tekstsequenties, volstaat het algemeen compressieschema uit hoofdstuk 2 niet langer om de discriminerende eigenschappen van compressiealgoritmen te onderscheiden. In tegenstelling tot tekst hoeft het bronobject niet langer eendimensionaal geïndexeerd te zijn. Maar dit object moet wel gestructureerd zijn, wat vervat is in de term *gestructureerde toevalsgrootheid* of kortweg *toevalsstructuur*. Het weze duidelijk dat niet de structuur zelf maar de gestructureerde grootheid een toevalskarakter heeft. Een *sequentie* en een *constellatie* zijn de uitkomst van respectievelijk een toevalsrij en een toevalsstructuur.

In figuur 4.4 stellen we een nieuw paradigma voor dat een conceptueel overzicht biedt van de opbouwende entiteiten van een compressiesysteem. Deze zienswijze is gebaseerd op [189] en benadert de verzameling van mogelijke dan wel typische constellaties vanuit een informatietheoretisch standpunt. Ze is maar één van de mogelijke zienswijzen die toelaat om zowel wetenschappelijk gefundeerde als ad hoc methoden op te delen. De scheiding van de entiteiten binnen het paradigma is in veel algoritmen niet altijd expliciet aanwezig of zelfs eenduidig. Bepaalde entiteiten kunnen anderzijds dan weer meermaals voorkomen binnen één techniek. Belangrijk is dat het toepassen van dit paradigma op de meeste statistische compressietechnieken verhelderend kan zijn en dat het toelaat een taxonomie op te stellen van de talloze bestaande technieken. De structurele en hybridische technieken zijn hier iets moeilijker in onder te brengen. Dit conceptueel schema bevat geen tijdsaspecten zodat het niet kan beschouwd worden als de visualisatie van een datapad. Maar indien onderscheid gemaakt wordt tussen de entiteiten van het algoritme kan het datapad gemakkelijk afgeleid worden.

Op het hoogste niveau onderscheiden we enerzijds het aspect van de *modelvorming* en anderzijds het aspect van de *codeassignatie*. Compressie heeft tot doel alle typische sequenties of constellaties in een kortere maar equivalente vorm voor te stellen. De entiteit *codeassignatie* is een essentieel onderdeel dat voor de voorstelling zorgt en de entiteit *modelvorming* zorgt er precies voor dat deze voorstelling korter is dan de oorspronkelijke voorstelling. Modelvorming en codeassignatie zijn de enige entiteiten die in elk compressiealgoritme expliciet aanwezig zijn. Indien bepaalde entiteiten een triviale vorm aannemen, zoals bijvoorbeeld de identieke functie of de uniforme distributie, worden ze



Figuur 4.4: Een nieuw paradigma dat de entiteiten voorstelt waaruit een compressieschema opgebouwd is.

als impliciet aanwezig verondersteld.

De modelvorming is opgebouwd uit een subentiteit *organisatie* en een subentiteit *parameters*. De organisatie bepaalt hoe we tegen de sequentie aankijken en heeft als doel de onderliggende structuur en haar probabilistische eigenschappen te ontleden. Uit de definitie van de organisatie volgt welke sequenties typisch zijn en welke atypisch. De parameters beschrijven de vrijheidsgraden die toelaten om een typische sequentie te onderscheiden van de andere typische sequenties.

De organisatie omhelst drie soorten acties: een *transformatie*, een *sorteerfunctie* en een *classificatie*.

- **transformatie**: De *transformatie* zet de sequentie om in een nieuwe sequentie met duidelijker aanwezige redundantie en is aldus verantwoordelijk voor het expliciet maken van de redundantie. Aan de decodeerzijde vindt de inverse transformatie plaats. Voor verliesloze compressie moet deze transformatie reversibel zijn. Voor verlieshebbende compressie is dit de actie die verantwoordelijk is voor het geïntroduceerde verlies. De transformatie kan mogelijks de onderliggende structuur veranderen en wordt *structuurbewarend* genoemd indien ze de structuur ongewijzigd laat. Compressietechnieken die veel interpretatie hechten aan de beeldinformatie brengen deze geavanceerde modellering doorgaans onder in de transformatie.

- **sorteerfunctie**: De *sorteerfunctie* beeldt de constellatie af op een sequentie waarbij een bijectieve relatie bestaat tussen de indexering van de onderliggende structuur en de natuurlijke getallen. Deze stap is essentieel omdat zowel een transmissiekanaal als een opslagruimte lineaire

media zijn, terwijl het bronobject arbitrair gestructureerd kan zijn. Om verliesloze reconstructie mogelijk te maken moeten de transformatie en de sorteerfunctie van die aard zijn dat ze de causaliteit niet in gedrang brengen.

- **classificatie**: De *classificatie* of de *structuurfunctie* beeldt de oneindigheid aan mogelijke verleden en toekomstige sequenties af op een eindig aantal klassen, respectievelijk *contexten* en *evoluties* genoemd. Doordat de onderliggende structuur topologisch kan verschillen van een rij, zal doorgaans niet alle nabuurschap bewaard blijven onder de sorteerfunctie. Om hiermee rekening te houden maakt de classificatie dikwijls zelf gebruik van een sorteerfunctie. Om een onderscheid te maken worden ze respectievelijk de *globale* en de *lokale* sorteerfunctie genoemd. De globale sorteerfunctie bepaalt de verwerkingsvolgorde van de bronwaarden, terwijl de lokale sorteerfunctie voor iedere bronwaarde een pad in het verleden en in de toekomst definieert. Meestal is ofwel de contextclassificatie ofwel de evolutieclassificatie triviaal en is slechts één ervan expliciet geïmplementeerd, maar dat hoeft niet zo te zijn.

Het toepassen van deze opdeling op een aantal bewerkingen is niet altijd eenduidig. Sommige bewerkingen zijn onder te brengen als een transformatie, maar ook als een classificatie van de toekomstige sequenties. Per conventie stellen we dat in dergelijke situaties gekeken wordt naar het lokale karakter van de bewerking. Is het resultaat van de bewerking afhankelijk van de waarnemingen in het volledige verleden, dan is het een classificatie. Is het resultaat daarentegen altijd hetzelfde onder identieke lokale situaties, dan is het een transformatie. Een mooi voorbeeld hiervan is het afbakenen van de overeenkomsten in LZ-gebaseerde technieken. Omdat deze bewerking expliciet voortbouwt op de waarnemingen uit het verleden, stellen we bij conventie dat ze als een classificatie van evoluties wordt beschouwd.

De parameters kunnen verder opgesplitst worden in drie soorten grootheden: *bronwaarden*, *statistieken* om de ermee geassocieerde waarschijnlijkheden te schatten en verder nog *zij-informatie*.

- **bronwaarden**: De *bronwaarden* stellen de mogelijke evoluties voor. De lijst van bronwaarden stelt de uitkomstenverzameling voor van het toevalsexperiment dat bestaat uit één cyclus van het algoritme.

- **statistieken**: De conditionele waarschijnlijkheidsdistributie van deze toevalsveranderlijke wordt empirisch geschat aan de hand van de *statistieken*. De voorstelling van bronwaarden en geassocieerde geschatte waarschijnlijkheden kan enumeratief of parametrisch gebeuren.

- **zij-informatie**: De *zij-informatie* omvat alle parameters en implementatiekeuzes die niet aan de entropiecoder van de codeassignatie doorgegeven worden. De lengte van de voorstelling van de zij-informatie is doorgaans klein of zelfs verwaarloosbaar.

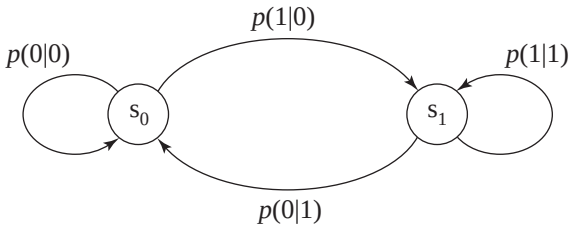
Doorgaans worden ofwel enkel de bronwaarden ofwel enkel de statistieken expliciet voorgesteld en is de ander triviaal, maar dat hoeft niet zo te zijn.

De codeassignatie kent codewoorden toe aan de bronwaarden en aan de zij-informatie en genereert op deze manier een machinevoorstelling voor de constellatie. Indien de bronwaarden niet uniform verdeeld zijn, wordt gebruik gemaakt van een entropiecoder zoals beschreven in hoofdstuk 3. Het resultaat van de entropiecoder wordt het bericht of de *gecodeerde bitstroom* genoemd.

Technieken die gebruik maken van gemengde modellen combineren meerdere modelvormingen tot één globale modelvorming. Hierbij treedt een extra parameter *gewicht* op binnen iedere modelvorming en zijn bepaalde entiteiten zoals de globale sorteerfunctie vaak gemeenschappelijk. De som van de gewichten over alle modelvormingen heen is 1. Dikwijls zijn de gewichten binair, waarbij één modelvorming als gewicht 1 heeft en het gewicht van alle andere modelvormingen samen 0 is.

Op deze manier bakent de modelvorming voor elk compressiealgoritme een verzameling typische constellaties af en is de codeassignatie van die aard dat de verwachte lengte over alle typische constellaties nagenoeg minimaal is. Dit paradigma laat toe de parallellen en de verschillen tussen de compressiealgoritmen op een consistente manier na te gaan. De grootste tekortkoming ervan is dat sommige technieken op meerdere manieren kunnen opgesplitst worden in de respectieve entiteiten. Het is niet onze bedoeling om elke techniek erop te projecteren, maar wel om het aan te halen daar waar het verhelderend werkt. Bij de beschrijving van contextmodellering van binaire beelden wordt bij wijze van voorbeeld deze terminologie toegepast op een concreet algoritme.

De opsplitsing van de hiernavolgende technieken is vooral gebaseerd op de structuur van de beeldinformatie nadat de transformatie is uitgevoerd. Bij looplengtecodering wordt de informatie voorgesteld aan de hand van rijen van natuurlijke getallen (in casu de looplengtes). Bij contextmodellering treedt er geen structuurwijziging op en is de transformatie de identieke functie. Bij patroongebaseerde technieken wordt de informatie voorgesteld aan de hand van een lijst die het optreden van bepaalde patronen beschrijft. Bij structurele technieken tenslotte zit de optimalisatie volledig vervat in de resulterende nieuwe structuur. Bepaalde officiële compressiestandaarden zijn evenwel ruimer gedefinieerd en combineren meerdere van deze technieken.



Figuur 4.5: Toestandstransitiediagram volgens het caponmodel voor binaire beelden.

4.3.2 Looplengtecodering

Looplengtecodering (*E.* run-length encoding) of RLE is een blokgebaseerde techniek die de inkomende sequentie opsplitst in blokken waar alle pixels '0' zijn en blokken waar alle pixels '1' zijn. Het onderliggend model gaat terug naar het *caponmodel*, een eenvoudig markovproces met twee toestanden [31]. Figuur 4.5 toont het toestandstransitiediagram met de toestanden s_0 en s_1, die arbitrair toegekend worden aan respectievelijk *zwart* en *wit*. Voor binaire beelden zijn de toestandstransitiewaarschijnlijkheden $p(0|0)$ en $p(1|1)$ veel groter dan $p(0|1)$ en $p(1|0)$. Deze waarschijnlijkheden beschrijven volledig het toevalskarakter van dit model.

Looplengtecodering is een mooi voorbeeld van een klasse compressietechnieken die op meerdere manieren binnen het voorgestelde paradigma passen. Het afbakenen van de looplengtes kan beschouwd worden als een transformatie, maar ook als een classificatie van mogelijke evoluties. Als transformatie beschouwd, beschrijft het afbakenen hoe een pixelrij omgezet wordt in een rij van natuurlijke getallen (looplengtes). Als classificatie beschouwd, legt het de klasse vast van mogelijke evoluties waarvoor een waarschijnlijkheid geschat wordt. De mogelijke evoluties zijn in dit geval de eindige evoluties van constante kleur inclusief de kleurwijziging. Omdat het bepalen van de looplengtes een lokale bewerking is, beschouwen we het als een transformatie, dit in tegenstelling tot LZ-gebaseerde technieken.

De ITU-T (Telecommunication Standardization Sector van de International Telecommunication Union, vroeger CCITT) heeft enkele *aanbevelingen* (*E.* recommendations) gepubliceerd voor faxcompressie die gebaseerd zijn op looplengtecodering [107]. *Groep 3* en *Groep 4* definiëren zowel de compressietechnieken als andere aspecten zoals communicatieprotocols. *Groep 3 (G3)* is gebaseerd op ITU-T aanbeveling T.4 en is ontworpen voor kanalen waar fouten kunnen optreden. *Groep 4 (G4)* is gebaseerd op ITU-T aanbeveling T.6 en veronderstelt een foutvrij transmissiekanaal. Sinds de oorspronkelijke

publicatie is de meeste functionaliteit van Groep 4 overgenomen in Groep 3. De codes zijn statisch en geoptimaliseerd voor een verzameling van typische documenten. Terzijde, Groep 1 en Groep 2 beschrijven analoge standaarden voor faxtransmissie.

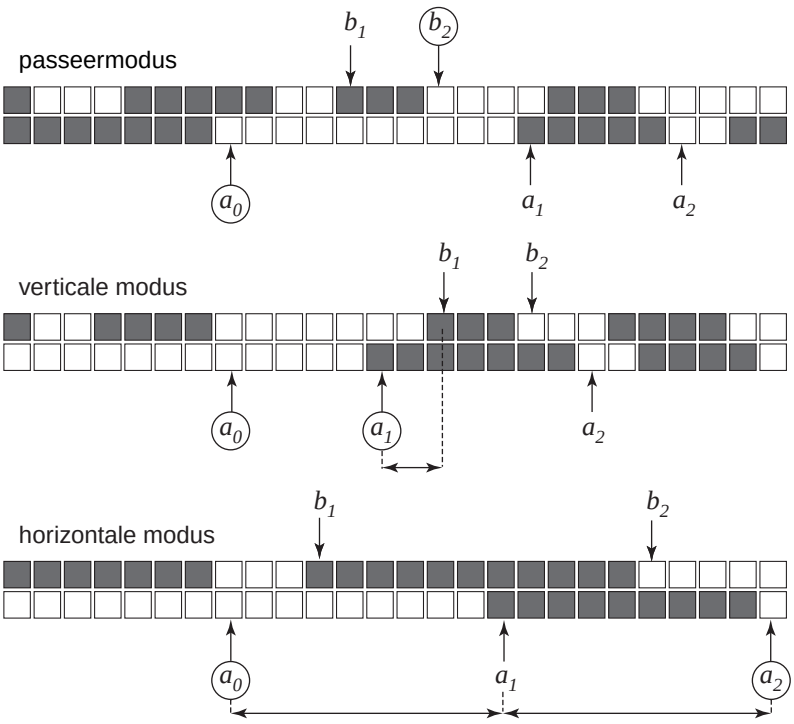
ITU-T Aanbeveling T.4 (G3)

Aanbeveling T.4 maakt gebruik van een eendimensionale en een tweedimensionale modus, respectievelijk gekend als MH en MR [112].

De eendimensionale modus, *Modified Huffman* of MH, maakt gebruik van statische gewijzigde huffmancodes om looplengtes l_0 en l_1 van respectievelijk zwarte en witte pixels binnen de huidige lijn te coderen. De gewijzigde huffmancode bestaat uit twee delen: een *opmaakcode* (*E.* make-up code) die $l_i \div 64$ codeert en een *afbreekcode* (*E.* terminating code) die l_i mod 64 codeert. Hierbij stelt $a \div b$ de gehele deling $\lfloor a/b \rfloor$ voor. De codewoorden verschillen voor zwarte en witte looplengtes.

De tweedimensionale modus, *Modified Relative Element Address Designate (Modified READ)* of MR, baseert zich op de voorgaande lijn om de relatieve posities van de kleurovergangen in de huidige lijn te bepalen en te coderen [107]. De huidige en vorige lijn worden respectievelijk de *codelijn* (*E.* coding line) en *referentielijn* (*E.* reference line) genoemd. Een *overgangspixel* is een pixel met de tegenovergestelde kleur als de vorige pixel (zijn linkerbuur). Volgende pixelposities worden gedefinieerd, zie figuur 4.6: a_0 is de laatst verwerkte pixel van de codelijn en zijn positie volgt uit elke codeerstap, a_1 en a_2 zijn respectievelijk de eerste en de tweede overgangspixel na a_0, b_1 is de eerste overgangspixel op de referentielijn rechts van a_0 en met dezelfde kleur als a_1 en b_2 is de eerste overgangspixel rechts van b_1.

Afhankelijk van de relatieve positie van deze pixels worden er drie modi onderscheiden: *passeermodus*, *verticale modus* en *horizontale modus*. Deze drie modi en de corresponderende pixelposities worden geïllustreerd in figuur 4.6. Met elk van deze modi is één codewoord geassocieerd die de modus aangeeft en eventueel ook een statische MH-code voor de looplengten. Indien b_2 links ligt van a_1, wordt het codewoord voor de *passeermodus* (*E.* pass mode) doorgestuurd. De pixel onder b_2 wordt de nieuwe positie van a_0. In de overige gevallen wordt gekeken naar de relatieve horizontale positie van a_1 ten opzichte van b_1. Is de horizontale afstand $|a_1b_1| \leq 3$, dan wordt deze afstand gecodeerd volgens het codeboek van de *verticale modus* (*E.* vertical mode) en wordt a_1 de nieuwe positie voor a_0. Is de horizontale afstand $|a_1b_1| > 3$, dan wordt het codewoord voor *horizontale modus* (*E.* horizontal mode) doorgestuurd, evenals de statische codewoorden voor $|a_0a_1|$ en $|a_1a_2|$ volgens het



Figuur 4.6: De drie modi gebruikt door MR. De streeplijnen en horizontale pijlen wijzen op de looplengtes. De oude en nieuwe lokatie van a_0 wordt aangegeven door de omcirkelde symbolen.

codeboek van de horizontale modus. De nieuwe positie voor a_0 wordt nu gegeven door a_2.

De methode beschikt over een aantal eigenschappen om de impact van transmissiefouten te verkleinen. Zo specificeert MR dat elke K-de rij (met K = 1, 2 of 4) gecodeerd wordt aan de hand van MH. Voor K = 1 betekent dit dat het volledige beeld in eendimensionale modus gecodeerd wordt. Voor normale resolutie is $K = 2$ en voor hoge resolutie kan $K = 4$ gebruikt worden. Bovendien beschikt MR nog over een aantal controlesymbolen voor de vlotte verwerking van het gecodeerd bericht. Aan het einde van een lijn zijn er ook voorzieningen voor opvulbits om de bytegrenzen te respecteren en lijnen worden van elkaar gescheiden door een ‘end-of-line’-symbool (‘`EOL`’).

ITU-T Aanbeveling T.6 (G4)

Aanbeveling T.6 maakt enkel gebruik van de tweedimensionale modus [18, 113]. Deze techniek stelt $K = \infty$ en wordt ook wel *Modified-Modified READ* of MMR genoemd. Er wordt een virtuele nulde lijn verondersteld die helemaal wit is zodat ook de eerste lijn in tweedimensionale modus kan gecomprimeerd worden. Daarenboven worden de controlekarakters achterwege gelaten. Dit is enkel mogelijk dankzij de veronderstelling dat het kanaal foutvrij is. De Europese faxtoestellen maken gebruik van G3-codering terwijl G4-codering gebruikt wordt voor de computervoorstelling van binaire beelden.

Beide aanbevelingen zijn volledig statisch. Een voordeel hiervan is dat de codering en de decodering zeer snel verloopt en eenvoudig in hardware is om te zetten. Een nadeel is dat de klasse van typische beelden zeer klein is. Deze technieken scoren dan ook bijzonder slecht op beelden met statistieken die sterk afwijken van die van tekstbeelden. Zo levert toepassing van G3-codering (met $K = 1$) op een damboordpatroon een expansie op met een factor 4.5, maar ook voor halftoonbeelden kunnen de gevolgen desastreus zijn. Dit is één van de problemen die opgelost worden in de nieuwe JBIG-faxstandaarden.

4.3.3 Contextmodellering

De aanpak waarbij de statistische modelvorming volledig gescheiden wordt van de codeassignatie ontstond kort na de ontwikkeling van aritmetische codes [189]. In verhouding tot alfabetuitbreidingtechnieken zoals uitgebreide huffmancodes is deze nieuwe aanpak inherent superieur. De nieuwe inzichten werden eerst concreet uitgewerkt voor binaire beelden en pas later aangewend voor tekstcompressie [136]. De reden hiervoor ligt in het binaire karakter van de pixels. Kort daarna stelde Rissanen voor hoe de contextfunctie (in casu de lengte van de context) kan bepaald worden aan de hand van een criterium voor de minimale conditionele entropie [191]. Onafhankelijk van deze vernieuwde denkbeelden over compressie ontstond de idee om statistische modellen van verschillende orde te mengen. Uit de combinatie van beiden ontstonden de PPM-technieken. De volgende paragrafen veronderstellen dat de pixels slechts $l = 2$ verschillende waarden kunnen aannemen, tenzij anders vermeld.

Eindigecontextmodel

Voor de modellering van een beeld wordt eerst de rasterscan toegepast op de indexering, wat toelaat de beeldelementen eendimensionaal te adresseren. Vervolgens kan vertrokken worden van een *probabilistische eindige automaat* (*E.* probabilistic finite state machine) of PFSM van de vorm $s_{i+1} = f(s_i, x_i)$ met

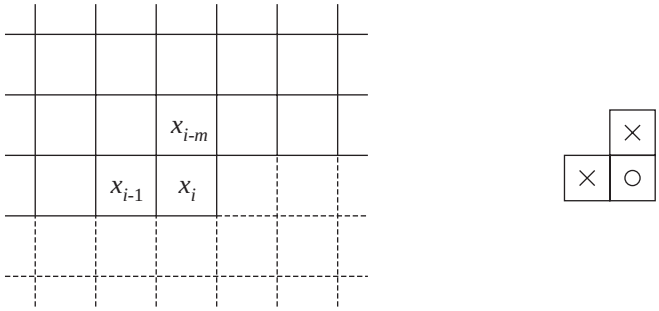
$s_0 = 0$ en probabiliteiten $p(x_i|s_i)$. Hierin is x_i de i-de pixel en is s_i de interne toestand waarin het systeem zich bevindt om x_i te modelleren. De functie f definieert de mogelijke toestandstransities en de respectieve probabiliteiten van deze transities zitten vervat in $p(x_i|s_i)$.

De toevalsrij bestaande uit de interne toestanden $\{S_i\}$ vormt een eerste-orde-markovproces. Maar dit betekent daarom nog niet dat ook de toevalsrij van de pixelwaarden $\{X_i\}$ markoviaans is. Een voldoende voorwaarde hiervoor is dat er een functie s en een orde r bestaat waarvoor de interne toestanden $s_i = s(x_{i-1}, x_{i-2}, \ldots, x_{i-r})$, met r een natuurlijk getal. De keuze voor deze *toestandsfunctie* s is arbitrair, maar de minst beperkende vorm voor s is de aaneenrijging $s_i = x_{i-1}^r = x_{i-r} \cdots x_{i-2}\, x_{i-1}$, geïnterpreteerd als binair getal. Iedere andere toestandsfunctie kan verfijnd worden tot deze vorm. De waarschijnlijkheden nemen de vorm $p(x_i|x_{i-1}^r)$ aan en er zijn $l^r(l-1)$ of dus 2^r vrijheidsgraden.

Is m de breedte van het beeld en bijgevolg ook de lengte van een lijn, dan moet r heel groot zijn opdat de interne toestanden de relevante beeldinformatie zouden bevatten. Immers, conditioneren op de pixel die k lijnen hoger ligt, vereist dat $r \geq km$. Het aantal vrijheidsgraden is onhandelbaar groot. Het invoeren van een *contextfunctie* g beperkt het aantal vrijheidsgraden door de interne toestand s_i af te beelden op een context $y_i = g(s_i)$ zodat $p(x_i|s_i) \approx p(x_i|y_i)$. Veelal maakt de contextfunctie g gebruik van een lokale sorteerfunctie. Is g van de vorm $g(x_{i-1}^r) = x_{i-m_1}\, x_{i-m_2} \cdots x_{i-m_q}$ voor een zekere $q \leq r$, dan vormen de relatieve posities $m_1, m_2, \ldots, m_q$ een *masker* of een *sjabloon* (*E.* template) voor de constructie van de contexten. Om causaliteitsredenen komen enkel pixelwaarden uit het verleden in aanmerking. Een sjabloon dat hieraan voldoet wordt een *causaal sjabloon* genoemd. De interne toestanden s_i worden soms de *microtoestanden* (*E.* micro states) genoemd, terwijl de externe toestanden of contexten y_i de *macrotoestanden* (*E.* macro states) zijn van het model. Contextmodellering met een statisch of semi-adaptief sjabloon van constante grootte en vorm wordt ook wel *sjablooncodering* (*E.* template coding) genoemd.

Het resulterende model voor $\{Y_i\}$ is niet langer een markovmodel maar een *verborgen markovmodel* (*E.* hidden markov model) of HMM. Het blijft echter een FSM en wordt een *eindigecontextmodel* (*E.* finite context model) of FCM genoemd. De *orde* q van een eindigecontextmodel voor beelden verwijst naar het aantal pixels in het contextsjabloon en niet naar de lengte r van de interne toestand. De overgang van tekst naar beelden heeft een aanpassing vereist van deze definities omwille van de gewijzigde topologie. Merk op dat deze aanpak niet langer opgaat bij de randen van een beeld.

Bij wijze van voorbeeld bespreken we eerst een eenvoudig contextmodel



Figuur 4.7: Een eenvoudig sjabloon dat gebruik maakt van twee contextpixels in uitgewerkte voorstelling (*links*) en in schematische voorstelling (*rechts*).

voor verliesloze compressie van binaire beelden. Figuur 4.7 stelt het sjabloon voor waarbij de aaneenrijging van de pixel links van de huidige pixel en de pixel boven de huidige pixel de context bepaalt. De pixels rechts van en onder de huidige pixel x_i mogen niet gebruikt worden omdat ze de toekomst voorstellen en bijgevolg is hun waarde nog niet gekend bij decompressie. Voor dit masker is $q = 2$, $r = m$, $s_i = x_{i-1}^m$ en dus is de toestandstransitiefunctie $f(x_{i-1}^m, x_i) = x_i^m$. De contexten worden gedefinieerd als $y_i = x_{i-1}\, x_{i-m}$ en bijgevolg is de contextfunctie $g(x_{i-1}^m) = x_{i-1}\, x_{i-m}$. De relatieve posities $m_1 = 1$ en $m_2 = m$ beschrijven volledig het sjabloon. Om tot het finale FCM te komen worden achtereenvolgens de veronderstellingen $p(x_i|x_{i-1}, x_{i-2}, \ldots, x_1) = p(x_i|x_{i-1}^r) = p(x_i|x_{i-1}\, x_{i-m})$ gemaakt. Het invoeren van een contextfunctie g herleidt het aantal vrijheidsgraden van 2^m naar 2^2. Dit eenvoudig contextmodel is een vereenvoudiging van het zeven of tien contextpixels tellende *LR-model* van Langdon en Rissanen [136].

Dit voorbeeld laat toe om de terminologie van het voorgestelde paradigma te illustreren. Elk binair beeld is een constellatie die als een matrix van de getallen '0' en '1' voorgesteld wordt. Hier treedt geen expliciete transformatie op, al zou er bijvoorbeeld een ruisonderdrukkend filter gehanteerd kunnen worden. De globale sorteerfunctie is de rasterscan die de lijnen één na één aaneenrijgt. De classificatie beeldt het volledige verleden x_{i-1}^{i-1} af op de context $x_{i-1}\, x_{i-m}$. Anders bekeken wordt de rij indices $i-1$, $i-2$, $\ldots$, $i-m$, $\ldots$ door de lokale sorteerfunctie afgebeeld op de rij $i-1$, $i-m$, $\ldots$ en wordt de orde beperkt tot twee. Verder laat de classificatie slechts twee mogelijke evoluties toe, namelijk '0' en '1', en deze vormen dan ook de impliciet aanwezige bronwaarden. De waarschijnlijkheden $\{p(t|uv);\ t, u, v \in \{0, 1\}\}$ worden expliciet geschat aan de hand van de telwaarden $\{\bar{c}(uv);\ u, v \in \{0, 1\}\}$ en $\{c(tuv);\ t, u, v \in \{0, 1\}\}$. Deze telwaarden en de eruit afgeleide geschatte

probabiliteiten $\hat{p}(t|uv)$ vormen de statistieken. De zij-informatie bestaat uit de breedte m en het aantal lijnen n van het beeld. De beste resultaten worden behaald indien een aritmetische coder gebruikt wordt als entropiecoder in de codeassignatie.

Het paradigma voor beeldcompressie is vooral interessant vanuit een veralgemenend en theoretisch standpunt. Het laat toe om de parallellen en verschillen tussen de voorgestelde technieken op een gestructureerde manier te analyseren. Het is evenwel geen must om elke techniek volgens deze zienswijze op te splitsen om de exacte werking ervan te doorgronden.

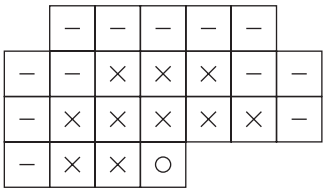
MG twee-niveau-contextmodel

Het "Managing Gigabytes" project of MG-project heeft tot doel een softwareplatform uit te bouwen voor de gelijktijdige compressie en indexering van teksten en beelden [273]. Het resultaat is dat de opslagruimte van een database samen met zijn indexering kleiner dan de helft is van de opslagruimte voor de ongecomprimeerde database. Bovendien verloopt het zoeken en ophalen vlugger door de uitgekiende interactie tussen de compressie en de indexering.

Het algoritme voor binaire beelden dat er voorgesteld wordt is een rechtstreekse uitbreiding van het LR-model. De vraag naar de optimale grootte en vorm van het sjabloon dringt zich op. Een grotere context geeft aanleiding tot een hogere efficiëntie op voorwaarde dat hij voldoende frequent voorgekomen is om betrouwbare statistieken af te leiden. Is dat niet zo, dan neemt de efficiëntie af. De optimale grootte van het sjabloon is bijgevolg niet alleen beeldafhankelijk maar ook contextafhankelijk. Omdat niet alle contexten evenveel voorkomen, worden sjablonen van twee afmetingen gecombineerd wat resulteert in een *twee-niveau-contextmodel* (*E.* two-level context model) [160]. Het sjabloon van figuur 4.8 stelt het standaard 22/10-sjabloon voor. Als de statistieken geconditioneerd op de grote context onvoldoende betrouwbaar zijn, valt het model terug op de kleine context. De implementatie MG-2L laat door de gebruiker gespecificeerde twee-niveau-sjablonen toe, maar voorziet zelf niet in de constructie van geoptimaliseerde sjablonen.

JBIG1

JBIG1 of kortweg JBIG is de laatste officieel verschenen standaard voor binaire beelden en is dan ook de opvolger van de ITU-T aanbevelingen T.4 en T.6 [6, 118]. De standaard is opgesteld door JBIG (Joint Bi-Level Image Processing Group), een groep die samenwerkt met ITU-T, IEC en ISO. De doelstellingen van dit compressieformaat zijn progressieve codering van binaire beelden, betere compressie voor gewone binaire beelden en het wegwerken

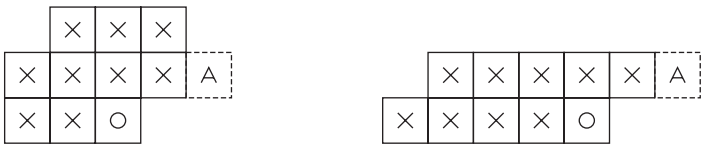


Figuur 4.8: Een 22/10-sjabloon voor een twee-niveau-contextmodel volgens het "Managing Gigabytes" project. De huidige pixel is gemarkeerd als '$\circ$'. Het kleine sjabloon bevat de 10 pixels gemarkeerd als '$\times$'. Het grote sjabloon omvat het kleine evenals de 12 additionele pixels gemarkeerd als '$-$'.

van de slechte resultaten voor gerasterde beelden. Het algoritme steunt op de volgende drie bouwblokken: progressieve codering, contextmodellering en de QM-coder.

Als eerste bouwblok gebruikt JBIG1 multiresolutiedecompositie om het coderen en decoderen progressief te laten verlopen. Daartoe wordt een beeld getransformeerd in een aantal *lagen* (*E.* layers). De laag met de laagste resolutie wordt de *bodemlaag* (*E.* bottom layer) of de *onderste laag* genoemd; de andere worden *verschillagen* (*E.* differential layers) of *hogere lagen* genoemd. Iedere laag telt in elke richting de helft van het aantal pixels van een hogere laag. Via een uitgekiende *onderbemonstering* (*E.* downsampling) wordt de lagere laag gegenereerd op basis van de hogere laag. Dit moet met de nodige voorzorg gebeuren om het weglaten van detail, het weglaten van gesimuleerde grijstinten en het invoeren van artefacten of ruis te vermijden. Daarom gebruikt de onderbemonstering een meerderheidscriterium dat rekening houdt met buurtpixels op zowel de hogere als de huidige laag. Daarenboven zijn enkele uitzonderingen voorzien om structuren te bewaren zoals randen, horizontale en verticale lijnen, periodische patronen en rasterelementen. Dit komt de kwaliteit van de voorlopige reconstructie ten goede. Deze voorlopige reconstructie is een doel op zich, want de progressiviteit van de methode heeft een negatieve invloed op de compressie. Merk op dat de volgorde van de lagen, waarbij de onderste laag de laagste resolutie heeft, tegengesteld is aan de volgorde die bij monochrome technieken gehanteerd wordt, waar de hoogste laag de laagste resolutie zal hebben.

De statistische modellering aan de hand van contexten over verschillende lagen vormt het tweede bouwblok. De contextmodellering is een eenvoudige uitbreiding van het LR-model en is verschillend voor de bodemlaag en de hogere lagen. In de bodemlaag bestaat het sjabloon enkel uit pixels binnen dezelfde laag. Figuur 4.9 toont het in de standaard voorgestelde drielijns- en



Figuur 4.9: JBIG1 schrijft voor de laag met de laagste resolutie een sjabloon van drie lijnen (*links*) voor en een sjabloon van twee lijnen (*rechts*). De positie van de contextpixel gemarkeerd als 'A' is adaptief in te stellen.

tweelijnssjabloon voor de bodemlaag. Dit laatste wordt enkel gebruikt om hogere snelheden of lagere geheugenvereisten te bereiken. De sjablonen voor elk van de hogere lagen bestaan uit de dichtste vier pixels in de lagere laag en zes nabije pixels uit de huidige laag. Op de bodemlaag zijn er 1024 mogelijke contexten. Omdat er vier mogelijke relatieve posities zijn voor een pixel in een hogere laag ten opzichte van een lagere laag, zijn er 4096 mogelijke contexten in elke hogere laag. Deze vier mogelijke relatieve posities worden ook soms de vier *spatiale fases* genoemd.

De voorgestelde sjablonen maken gebruik van een *adaptieve sjabloonpixel* (*E.* adaptive template pixel) of *AT-pixel*. De locatie van deze pixel is volledig adaptief en kan zelfs gedurende de beeldcodering wijzigen. Als mogelijke techniek voor het bepalen van de AT-pixel stelt de standaard voor om de best gecorreleerde pixel binnen de huidige lijn op voldoende korte afstand te nemen. Deze functionaliteit werd ingevoerd om de slechte resultaten op halftoonbeelden weg te werken.

Daarnaast beschrijft JBIG1 een drietal optionele verbeteringen. Ten eerste is er *deterministische voorspelling* (*E.* deterministic prediction). Soms is het mogelijk een pixelwaarde in een verschillaag met zekerheid te voorspellen op basis van de reeds gekende pixels in die laag en de pixelwaarden in de lagere laag. Indien bijvoorbeeld drie pixels in de verschillaag '0' zijn en de overeenkomstige pixel in de lagere laag is '1', dan moet de vierde pixel in de verschillaag ook '1' zijn. Deterministische voorspelling detecteert al deze voorspelbare situaties en verwijdert deze pixels uit het codeerproces. Ten tweede is er *typische voorspelling in de verschillaag* (*E.* differential-layer typical prediction). Typische voorspelling is gebaseerd op de aanwezigheid van grote homogene gebieden. In een verschillaag is het aannemelijk dat als een lagereresolutiepixel identiek is aan zijn acht buren binnen die lagere laag, de vier ermee corresponderende pixels in de huidige laag identiek zullen zijn aan die lagereresolutiepixel. Gaat deze eigenschap op voor een hele lijn op de lagere laag, dan codeert typische voorspelling in de verschillaag dit voor de twee

corresponderende lijnen van de huidige laag. Er wordt vooral een snelheidswinst geboekt. Ten derde is er *typische voorspelling in de bodemlaag*. Is een lijn identiek aan de vorige lijn, dan wordt deze lijn als typisch bestempeld. Typische voorspelling in de bodemlaag zoekt deze lijnen op en codeert hun aanwezigheid. Hier wordt vooral een winst in compressie geboekt.

Het derde bouwblok van JBIG1 wordt gevormd door de QM-coder. Dit is een aritmetische coder die geoptimaliseerd is voor binaire toevalsgrootheden, zie paragraaf 3.4.6. De vermenigvuldigingen zijn verdwenen en probabiliteiten voor het minder waarschijnlijke symbool (less probable symbol of LPS) worden gediscretiseerd tot 113 toestanden [68]. De overgangen tussen deze toestanden worden niet berekend maar gebeuren aan de hand van een toestandstransitietabel. De probabiliteitsschatting wordt op deze manier geïmplementeerd aan de hand van optellingen en een FSM. De QM-coder is dan ook meer dan een zuivere entropiecoder. De probabiliteitsschatter

$$\hat{p}(x|y^k) = \frac{c(y^k x) + \delta}{\bar{c}(y^k) + \delta},$$

met $\delta = 0.45$ ligt aan de basis voor het opstellen van de toestandstransitietabel. Adaptiviteit van de probabiliteitsschattingen zit ingebakken in de transities.

JBIG1 is ook in staat om grijswaardenbeelden en monochrome beelden te comprimeren. Daartoe wordt eerst de entropie van de bitvlakken verlaagd door graycodering toe te passen op de pixelintensiteiten. Deze reversibele transformatie kan eenvoudig geïmplementeerd worden als $x \oplus (x \gg 1)$, waarbij "$\oplus$" de binaire XOR-functie voorstelt en "$\gg$" de bitschuifoperatie (*E.* bit shift).

De standaard definieert enkel het decoderen en laat enige vrijheid toe wat het coderen betreft. Binnen de standaard zijn de vereisten beschreven waaraan een *basisimplementatie* (*E.* baseline implementation) moet voldoen. De standaard laat immers nog teveel vrijheid over om alle mogelijke implementaties te bundelen.

4.3.4 Patroongebaseerde technieken

Een techniek als JBIG1 is louter gebaseerd op contextmodellering en maakt abstractie van de eigenlijke beeldinhoud. In veel gevallen stelt het beeld een stuk tekst, een tekening of een gerasterde natuurlijke scène voor. Het beeld is dan opgebouwd uit een beperkt aantal basispatronen die zich op een gestructureerde manier herhalen. Deze basispatronen zijn dan gerasterde lettertekens of halftoondots. De contextsjablonen strekken zich slechts over enkele pixels uit en zijn dan ook onvoldoende groot om relevante statistieken op te bouwen voor de basispatronen. *Patroongebaseerde* (*E.* pattern-based) technieken hebben tot

doel deze basispatronen af te leiden en ze te bundelen tot een grafisch woordenboek. Vervolgens dient dit woordenboek als opzoektabel om het beeld op te splitsen en te coderen.

Een patroongebaseerde techniek combineert een structurele aanpak op beeldniveau met contextmodellering op pixelniveau. Om deze reden kan het beschouwd worden als een hybridische aanpak die de brug vormt tussen de zuivere contextmodellering uit de vorige paragraaf en de zuivere structurele technieken uit de volgende.

TIC — compressie van tekstbeelden

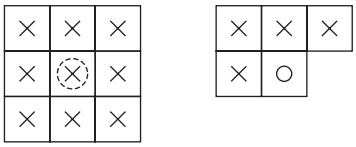
Het "Managing Gigabytes" project omvat een techniek geoptimaliseerd voor de compressie van *tekstbeelden* (*E.* textual images) [108, 273, 275]. Tekstbeelden zijn beelden die nagenoeg volledig bestaan uit gedrukte tekst. Andere elementen die kunnen optreden zijn logo's, handtekeningen en speciale symbolen. De techniek is eenvoudig uitbreidbaar naar bijvoorbeeld muziekpartituren omdat deze eveneens opgebouwd zijn uit een beperkt aantal terugkerende grafische basiselementen.

De voorgestelde techniek, *Textual Image Compression* of TIC, bouwt een woordenboek van letterpatronen op dat vervolgens gebruikt wordt om het beeld verliesloos of verlieshebbend te comprimeren. Het alternatief bestaande uit *tekstherkenning* (*E.* Optical Character Recognition) of OCR in combinatie met tekstcompressie is onhaalbaar omdat tekstherkenning niet in staat is de reproduceerbaarheid van alle elementen te garanderen.

Het algoritme codeert het beeld in twee fases. Een eerste fase construeert het woordenboek en codeert het beeld aan de hand hiervan op een verlieshebbende manier. De tweede fase gebruikt deze voorlopige reconstructie om via contextmodellering tot verliesloze compressie over te gaan.

De eerste fase heeft dus tot doel om een voorlopige reconstructie op te bouwen en omvat de volgende vier stappen.

- Zoek, isoleer en extraheer alle *tekens* (*E.* marks), dit zijn geconnecteerde gebieden.
- Middel gelijkaardige tekens uit tot één symbool en bouw een woordenboek op met deze gemiddelde symbolen.
- Zoek voor elk teken in het beeld het corresponderende symbool in het woordenboek en bepaal de *positiewijziging* (*E.* offset) tussen opeenvolgende tekens.
- Comprimeer het woordenboek, de symboolsequentie en alle positiewijzigingen.



Figuur 4.10: Het sjabloon bestaat uit een helderziend sjabloon actief in de voorlopige reconstructie (*links*) en een causaal sjabloon actief in het origineel beeld (*rechts*).

De patronen in het woordenboek worden gecomprimeerd aan de hand van een twee-niveau-contextmodel. Voor de symboolsequentie wordt PPMC gebruikt. De positiewijzigingen worden gecodeerd aan de hand van een orde-1 PPM-model.

Aan de decodeerkant volstaat deze informatie om een verlieshebbende reconstructie door te voeren. Indien deze voorlopige reconstructie volstaat, worden alle symbolen in het woordenboek opgeslagen. In het andere geval worden de symbolen die zelden voorkomen uit het woordenboek verwijderd, en volgt er een tweede fase.

De tweede fase bouwt verder op het resultaat van de eerste fase om verliesloze compressie te bereiken. Het binair coderen van het residubeeld, dit is het verschilbeeld tussen origineel en verlieshebbende reconstructie, lijkt een logische manier om dit te doen. Maar het is opmerkelijk dat het residu moeilijker te comprimeren is dan het originele beeld. Daarom wordt niet het residu maar het oorspronkelijk beeld gecomprimeerd aan de hand van contextmodellering. De contexten baseren zich zowel op het verleden van het oorspronkelijke beeld als op het verleden en de toekomst van de voorlopige reconstructie. Een sjabloon dat pixels uit de toekomst in rekening brengt lijkt niet-causaal en wordt daarom een *helderziend sjabloon* (*E.* clairvoyant template) genoemd, zie figuur 4.10.

JBIG2

Op het moment van schrijven van dit werk wordt JBIG2 voorgesteld, de meest recente compressiestandaard voor binaire beelden [120]. De belangrijkste wijziging ten opzichte van JBIG1 is dat tekstbeelden en halftoonbeelden op een hoger niveau van abstractie geïnterpreteerd en gemodelleerd worden. De benadering van de beeldinformatie leunt dichter aan bij de wijze waarop mensen deze informatie waarnemen. Deze hogere graad van interpretatie geeft aanleiding tot betere verliesloze compressie. Daarenboven zijn er meerdere voorzieningen getroffen voor gecontroleerde verlieshebbende compressie. Hierbij is het de bedoeling dat de ingevoerde artefacten onder normale omstandig-

heden niet zichtbaar worden. De definitieve versie is nog maar pas gepubliceerd [89, 99, 120].

De standaard beschrijft expliciet het formaat van de gecodeerde bitstroom en impliciet de implementatie van de decoder. Hoe de bitstroom gegenereerd wordt, ligt niet vast. De standaard voorziet in voldoende ontwerpvrijheid zodat meerdere commerciële implementaties met elkaar kunnen wedijveren.

Bovendien moet de standaard inzetbaar zijn in een breed gamma aan toepassingen, variërend van eenvoudige en trage faxtoepassingen tot zeer efficiënte en complexe draadloze transmissie. De beschikbare rekenkracht en het vereiste geheugen lopen sterk uiteen. Daarom zijn er in tegenstelling tot JBIG1 geen vereisten opgegeven voor een basisimplementatie, maar worden er meerdere *toepassingsprofielen* (*E.* application profiles) gespecificeerd.

De datasegmenten die samen de bitstroom vormen, zijn van verschillende types: *beelddatasegmenten* (*E.* image data segments) die de interne samenstelling van de beelden beschrijven, *woordenboekdatasegmenten* (*E.* dictionary data segments) die de opbouwende elementen van die beelden beschrijven en *controledatasegmenten* (*E.* control data segments) die structurele informatie bevatten zoals de paginabeschrijving. Documenten van meerdere pagina's kunnen woordenboekdatasegmenten delen en dit komt zowel de snelheid als de efficiëntie ten goede. De bitstroom heeft een controlestructuur die dergelijke documenten ofwel sequentieel toegankelijk ofwel direct toegankelijk (*E.* random access) maakt. In het laatste geval wordt alle controle-informatie in het begin van het bestand gebundeld. De bitstroom kan ingebed worden in andere formaten zoals TIFF (Tagged Image File Format).

Progressieve reconstructie kan op twee manieren plaatsvinden: *kwaliteitsprogressieve* (*E.* quality-progressive) reconstructie waarbij de progressie verloopt van lage naar hoge kwaliteit en *inhoudsprogressieve* (*E.* content-progressive) reconstructie waarbij achtereenvolgens de verschillende beeldtypes gedecodeerd worden. De kwaliteitstoename wordt uitgevoerd door *verfijningscodering* (*E.* refinement coding), doorgaans is dit een contextmodel gekoppeld aan een aritmetische coder.

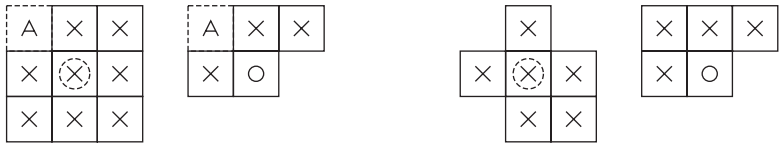
Het uitgangspunt voor JBIG2 is dat voor JBIG1 de typische samenstelling van binaire beelden niet weerspiegeld wordt in de modelvorming. Een sjabloon dat slechts enkele pixels ver kijkt volstaat niet om de tekstkarakters of de halftoondots volledig te vatten. De kracht van JBIG2 is precies de onderstelling dat de binaire beelden tekst of halftooninformatie voorstellen. Andere beelddata zoals lijnwerk wordt gecodeerd met een *opkuiscoder* (*E.* cleanup coder), hiervoor kan JBIG1 of MMR gebruikt worden. De opdeling van een pagina in de verschillende datatypes, ook wel de *segmentatie* (*E.* segmentation) genoemd, is een taak die niet beschreven wordt door de standaard maar volledig over-

gelaten wordt aan de coder. Als entropiecoder wordt de MQ-coder gebruikt. Deze steunt op dezelfde principes als de QM-coder van JBIG1, maar heeft enkele interessante eigenschappen voor de creatie van bitstromen. Bovendien is deze vrij te gebruiken binnen de ISO-standaardisatie.

Tekstbeelden De techniek die tekstbeelden modelleert vertoont veel gelijkenissen met de aanpak van TIC. In een eerste stap wordt het tekstbeeld gesegmenteerd in individuele tekens die hier *pixelblokken* (*E.* pixel blocks) worden genoemd. Vervolgens wordt elk pixelblok gecodeerd aan de hand van een woordenboek van pixelblokken op één van volgende manieren:

- **PM&S** (Pattern Matching and Substitution): Er wordt een "goede" *overeenkomst* (*E.* match) gezocht tussen het huidige pixelblok en de pixelblokken van het woordenboek. Wordt een dergelijke overeenkomst gevonden, dan wordt de index van de overeenkomst in het woordenboek gecodeerd. In het andere geval wordt het pixelblok gecodeerd met een opkuiscoder en wordt het pixelblok toegevoegd aan het woordenboek. Deze techniek is inherent verlieshebbend omdat de overeenkomst tussen het huidige pixelblok en het pixelblok uit het woordenboek nagenoeg nooit exact is. In het slechtste geval kunnen zelfs *vervangingsfouten* (*E.* substitution errors) optreden, zoals bijvoorbeeld wanneer een letter '*t*' als een '*l*' gelezen wordt.

- **SPM** (Soft Pattern Matching): SPM gaat nog een stap verder dan PM&S en maakt verliesloze codering mogelijk. Net zoals PM&S codeert deze techniek in eerste instantie de index van de overeenkomst. Maar daarna volgt een verfijningscodeerstap die, voortbouwend op de overeenkomst uit het woordenboek, het waargenomen pixelblok exact codeert. Indien gewenst wordt het zopas waargenomen pixelblok toegevoegd aan het woordenboek. De verfijningscodering gebeurt aan de hand van een contextmodel gekoppeld aan een aritmetische coder. Het sjabloon van het contextmodel is samengesteld uit enerzijds een helderziend sjabloon actief in het overeenkomstig pixelblok uit het woordenboek en anderzijds een causaal sjabloon actief in het waargenomen pixelblok. Elk van deze subsjablonen kan één AT-pixel bevatten. Figuur 4.11 toont de twee samengestelde sjablonen van verschillende afmetingen die in de standaard zijn opgenomen. Indien de overeenkomst verkeerdelijk als "goed" beoordeeld wordt, geeft dit wel aanleiding tot slechtere compressie maar niet tot vervangingsfouten. In tegenstelling tot PM&S kan deze techniek verliesloos werken, al hoeft dat niet noodzakelijk zo te zijn.



Figuur 4.11: SPM combineert een herderziend en een causaal sjabloon. Het samengesteld sjabloon kan 13 pixels tellen waarvan er 2 adaptief zijn (*links*) of kan uit enkel 10 vaste pixels bestaan (*rechts*).

Onafhankelijk van het gevolgde schema moet de *positiewijziging* (*E.* offset) van elk pixelblok ten opzichte van het vorige pixelblok gecodeerd worden. De woordenboeken van pixelblokken behoren tot de woordenboekdatasegmenten, terwijl de positiewijzigingen evenals de verwijzingen naar het woordenboek bij de beelddatasegmenten gerekend worden. De interpretatie van de tekst is font-, taal- en karaktersetonafhankelijk en is daarom veel robuuster dan tekstherkenning (OCR).

Halftoonbeelden Eén van de vooropgestelde doelen van de nieuwe JBIG2-standaard is betere compressie van halftoonbeelden [148, 149]. De aanpak van JBIG1, contextmodellering, werd verfijnd en er werd een tweede aanpak opgenomen die gebaseerd is op invers rasteren.

- **contextmodellering**: Een eerste aanpak is een uitbreiding van de sequentiële contextmodellering in JBIG1. Er zijn vier sjablonen toegelaten. De kleinste twee zijn dezelfde als de sjablonen van grootte 10 in JBIG1, zie figuur 4.9. De andere twee zijn nieuw en tellen respectievelijk 13 en 16 pixels, waarvan er respectievelijk 1 en 4 adaptief zijn. Figuur 4.12 geeft deze sjablonen weer. Een vuistregel stelt dat de grootste sjablonen bij voorkeur ingezet worden voor de grootste beelden. Contexten van tussenliggende grootte kunnen gesimuleerd worden door AT-pixels te laten samenvallen met vaste pixels.

- **invers rasteren**: De omgekeerde stap van rasteren, waarbij een grijswaardenbeeld gegenereerd wordt uit een halftoonbeeld, wordt *invers rasteren* of *ontrasteren* (*E.* descreening) genoemd. De voorgestelde aanpak is gericht op klassieke halftoonbeelden en laat geen verliesloze reconstructie toe. Een rechthoekig rooster geroteerd over een bepaalde hoek verdeelt het binair beeld in pixelblokken die overeenstemmen met de halftoondots. Bij ieder pixelblok hoort een grijswaarde, bijvoorbeeld het aantal zwarte pixels, en op deze wijze ontstaat een grijswaarden-

Figuur 4.12: JBIG2 voegt twee grotere sjablonen toe aan die van JBIG1. Er is een sjabloon van 16 pixels waarvan 4 adaptief (*links*) en een sjabloon van 13 pixels waarvan 1 adaptief (*rechts*).

beeld. Net zoals in JBIG1 wordt het grijswaardenbeeld gecomprimeerd door achtereenvolgens graycodering toe te passen op de grijswaarden en contextgebaseerde aritmetische codering toe te passen op de bitvlakken. Om het halftoonpatroon te bewaren, stemt met elke grijswaarde een pixelblok overeen in een woordenboek van halftoondots. Er worden geen veronderstellingen gemaakt qua dotvorm en dotgrootte. Een nuttige eigenschap van deze aanpak is dat decompressie op deze manier heel snel verloopt [79]. Compressie gebaseerd op invers rasteren is dan ook gelijkaardig aan de hierboven besproken technieken SPM en PM&S. Merk op dat ook hier enkel het decodeeralgoritme gespecificeerd wordt en dat de standaard dan ook veel keuzevrijheid overlaat voor de codering, in casu het bepalen van de vrijheidsgraden zoals de rasterhoek en halftoonvorm.

Voor- en nabewerken De standaard beschrijft op een eenduidige manier hoe een correct geformatteerde bitstroom gedecodeerd kan worden tot een binair beeld. Bepaalde codeertechnieken, zoals PM&S en invers rasteren, mogen dan wel inherent verlieshebbend zijn, eenzelfde gecomprimeerd bestand geeft toch altijd aanleiding tot hetzelfde gereconstrueerd beeld. Daarenboven kan het wenselijk zijn om tijdens het coderen of na het decoderen extra stappen uit te voeren om de efficiëntie te verhogen of de beeldkwaliteit te verbeteren. Deze stappen hebben als gevolg dat de waarde van een beperkt aantal pixels wijzigt zonder dat de globale impressie van het beeld beduidend verandert. Als *voorbewerking* (*E.* preprocessing) komen onder andere de volgende stappen in aanmerking: kwantisatie van de positiewijzigingen, ruisonderdrukking door het verwijderen van zeer kleine pixelblokken, afvlakken van de gemiddelde pixelblokken, uitlijnen van randen en het omklappen van onwaarschijnlijke individuele pixels. Vooral deze laatste stap dient met de nodige terughoudendheid uitgevoerd te worden omdat er een lawine-effect kan optreden met zichtbare artefacten als gevolg. Alhoewel deze in de standaard niet beschreven

wordt, kan *nabewerking* (*E.* postprocessing) in bepaalde situaties een gunstig effect hebben op de uiteindelijke beeldkwaliteit. Eén concrete situatie waar dit wenselijk is, is herbemonstering naar een nieuw uitvoermedium. Deze voor- en nabewerkingsstappen laten toe om van JBIG2 een gecontroleerde verlieshebbende techniek te maken.

4.3.5 Structurele technieken

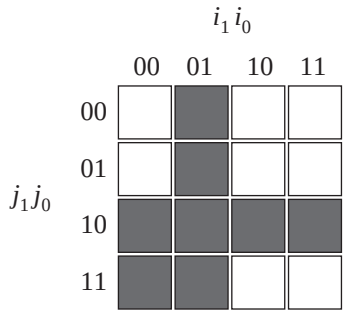
Structurele technieken zetten het beeld om in een datastructuur die volledig afhankelijk is van het beeld. Het uitbuiten van de redundantie in het beeld gebeurt door de grootte en de complexiteit van deze datastructuur te minimaliseren. Deze optimalisatietechnieken vinden hun oorsprong in andere takken van de wetenschap. Bij een voor de hand liggende implementatie komen weinig of geen statistieken of geassocieerde entropiecodes voor.

De technieken voor het optimaliseren van de datastructuur vinden hun oorsprong in de schakelalgebra en de digitale synthese bij VLSI-ontwerp. Daar bestaat immers een gelijkaardige zoektocht naar de meest efficiënte beschrijving van een vooraf opgesteld gewenst gedrag. De lengte van de geoptimaliseerde voorstelling is bij compressie een maat voor de efficiëntie en bij circuitontwerp een maat voor de kost van de hardware-implementatie. De grondgedachte is dat een binair beeld geïnterpreteerd wordt als een Boolese functie waarbij de pixeladressering de ingang voorstelt van de functie en de pixelwaarde de uitgang. Na de eigenlijke optimalisatie van de structuur moet ook het resultaat van deze stap zo efficiënt mogelijk gecodeerd worden. Dit betekent dat de compressie voor een stuk verschoven is weg van de oorspronkelijke beeldstructuur naar een nieuwe vrije structuur. Het is dan ook niet evident om een optimale beschrijvingslengte van een geminimaliseerde datastructuur vast te leggen.

De voorgestelde methoden vertonen veel overeenkomst maar verschillen fundamenteel op een aantal vlakken. We onderscheiden technieken gebaseerd op twee- en vierbomen, op Boolese minimalisatie, op geordende binairebeslissingsdiagrammen en op eindige automaten.

Twee- en vierbomen

Technieken gebaseerd op vierbomen zijn heel eenvoudig en zijn bovndien de best gekende. Een *vierboom* (*E.* quadtree) is een boomstructuur waarbij elke tussenknoop exact vier kindknopen heeft overeenkomend met de vier kwadranten van een opgesplitst vierkant [210]. Elke eindknoop of blad stelt de kleur wit of zwart voor en het gehele beeld stemt overeen met de wortel van



Figuur 4.13: Een eenvoudig binair testbeeld met pixeladressering die als ingang van een Boolese functie beschouwd wordt. De pixelintensiteit definieert de uitgang (‘1’ voor zwart en ‘0’ voor wit).

de boom. Om de kleur van een pixel te bekomen volstaat het de boom te doorlopen vertrekkend van de wortel en telkens de tak te kiezen die overeenkomt met het gekozen kwadrant bij elke resolutieverdubbeling.

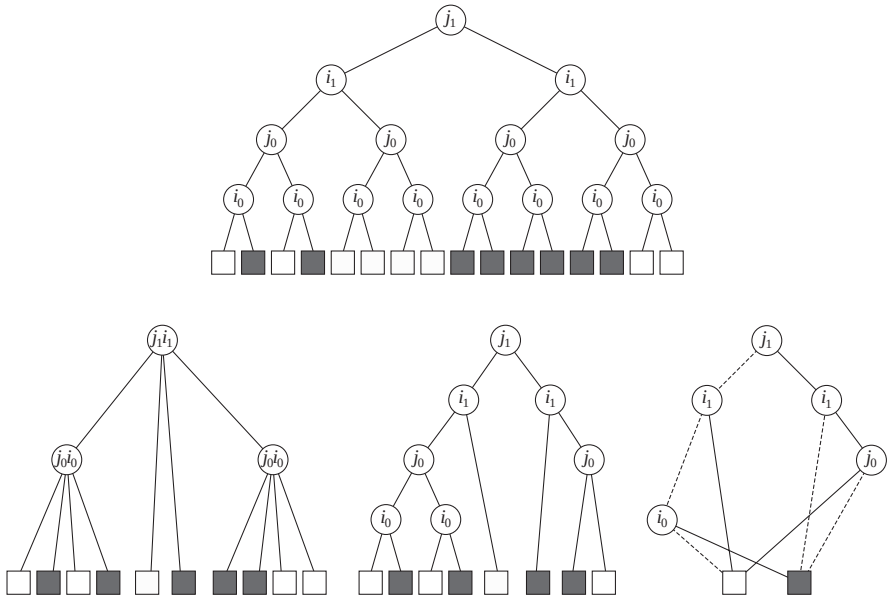
Een *tweeboom* (*E.* bintree) is gelijkaardig maar elke tussenknoop heeft slechts twee kindknopen. Het aantal lagen van de boom is gemiddeld dubbel zo groot en alternerend komt een resolutieverdubbeling in verticale dan wel in horizontale richting aan bod. Voor typische beelden is er weinig inherent verschil tussen beide voorstellingen.

Beide datastructuren lenen zich uitstekend tot het efficiënt voorstellen van ijle beelden en van beelden met grote egale vlakken, maar schieten tekort voor beelden met veel detail of terugkerende patronen. Zo vormt een dambordpatroon het slechtst mogelijke beeld en resulteren beide voorstellingen in expansie in plaats van compressie. Figuur 4.13 geeft een testbeeld weer met de voorgestelde pixeladressering in beide richtingen. Figuur 4.14 illustreert de volledig uitgevouwen boom van dit testbeeld en de equivalente twee- en vierboom.

Boolese minimalisatie

Tweebomen en vierbomen beschrijven een te kleine structuurruimte om efficiënt te zijn over een grote klasse binaire beelden. Het doortrekken van de parallel tussen een digitaal beeld en een te minimaliseren Boolese functie creëert een veel ruimer kader en laat toe aanzienlijk betere resultaten te bekomen. De schakelalgebra kent een aantal algoritmen voor de optimalisatie van deze functies aan de hand van vooraf gestelde randvoorwaarden en kostcriteria.

Wordt de optimalisatie beperkt tot een tweeniveaurepresentatie, dan leidt



Figuur 4.14: Vertrekkend van de volledig uitgevouwen tweeboom (*boven*) voor het beeld uit figuur 4.13 kan een gereduceerde vierboom (*onder links*), een gereduceerde tweeboom (*onder midden*) of een OBDD (*onder rechts*) geconstrueerd worden. Bij deze laatste duidt de streeplijn en de volle lijn op respectievelijk de '0'- en de '1'-tak. Voor de andere bomen zijn de takken lexicografisch geordend.

het algoritme van Quine-McCluskey tot een minimaal aantal priemimplicanten [46]. Deze aanpak kan gebruikt worden om een binair beeld verliesloos te comprimeren [213]. De eigenlijke Boolese minimalisatie wordt er uitgevoerd door het pakket "ESPRESSO" of "KISS". Een uitbreiding van deze methode deelt het blok op in kleine rechthoeken van 4×8, minimaliseert deze elk afzonderlijk en genereert vervolgens een huffmancode over deze geminimaliseerde vormen [8, 33, 44]. Het graycoderen van de pixelindices bewaart de spatiale nabuurschap bij minimalisatie en levert doorgaans betere resultaten op. Tabel 4.3 geeft de gedragsbeschrijving en de geminimaliseerde vorm weer van de Boolese functie volgens het beeld in figuur 4.13. Hierbij worden de pixelindices $j_1 j_0$ en $i_1 i_0$ eerst onafhankelijk graygecodeerd, zodat het beeld als een *karnaughkaart* kan geïnterpreteerd worden.

De aanpak heeft een aantal fundamentele tekortkomingen. Enerzijds neemt de duur van het optimalisatiealgoritme exponentieel toe als functie van het aantal producttermen. Dit heeft als gevolg dat het beeld moet gesegmenteerd worden wat een afbreuk van de optimaliteit met zich meebrengt. Anderzijds heeft

Tabel 4.3: Boolese representatievormen van de functie voorgesteld in figuur 4.13 voor en na minimalisatie volgens een tweeniveaurepresentatie. Op de indices $j_1 j_0$ en $i_1 i_0$ wordt de graycode $g(\cdot)$ toegepast om het nabuurschap te bewaren. De notatie "x" stelt een "don't care" voor en de uitgangskolom na minimalisatie is per definitie '1', vandaar de ronde haakjes (x_{ij}).

VOOR minimalisatie			NA minimalisatie		
ingang		uitgang	ingang		(uitgang)
$g(j_1 j_0)$	$g(i_1 i_0)$	x_{ij}	$g(j_1 j_0)$	$g(i_1 i_0)$	(x_{ij})
00	00	0	11	xx	(1)
00	01	1	xx	01	(1)
00	10	0	1x	0x	(1)
00	11	0			
01	00	0			
...		...			
00	00	0			

de randvoorwaarde van een tweeniveaurepresentatie weinig betekenis in compressietermen. Een normaal damboordpatroon met graygecodeerde pixelindices is equivalent aan een niet te reduceren Boolese functie. Minimalisatie naar een meerniveaurepresentatie kent deze beperking niet, maar deze optimalisatie verloopt nog veel trager en is in de praktijk onhaalbaar. Bovendien zou het niet triviaal zijn om het resultaat van deze laatste optimalisatie efficiënt te coderen.

De aanpak van de onbegrensde Boolese minimalisatie leunt dicht aan bij het abstract concept van de Kolmogorov-complexiteit of de beschrijvende complexiteit. De universele computer wordt hier geïdentificeerd met een generisch binair digitaal netwerk.

Geordend binairebeslissingsdiagram (OBDD)

Een gelijkaardige aanpak is gebaseerd op een efficiënte uitbreiding van tweebomen. Een *geordend binairebeslissingsdiagram* (*E.* ordered binary decision diagram) of OBDD is een diagram waarbij elk knooppunt een Boolese veranderlijke voorstelt. De eindknopen stellen het aantal mogelijke uitkomsten voor, twee in het geval van binaire beelden. Iedere tussenknoop heeft twee takken, één voor het geval de veranderlijke de waarde '0' aanneemt (streeplijn) en één voor de waarde '1' (volle lijn). Het diagram is geordend volgens de pixelindexering.

De kracht van een techniek die gebaseerd is op OBDD's situeert zich in

het bestaan van optimalisatiealgoritmen waarvan de tijdsduur logaritmisch oploopt als functie van de afmetingen. Bovendien is deze aanpak geschikt om terugkerende patronen die het binair karakter van de pixeladressering volgen te detecteren. De aanpak faalt dan ook niet op gedetailleerde repetitieve structuren zoals een dambordpatroon. Rechtstreekse optimalisatie zonder aandacht voor de codering van de optimale OBDD leidt tot een compressietechniek met beperkte efficiëntie [231]. In combinatie met graycodering van de indexering en entropiecodering van de optimale OBDD volgens een aritmetische code worden aanzienlijk betere resultaten behaald [150]. Figuur 4.14 illustreert een minimale OBDD met een beperkt aantal knooppunten voor het testbeeld van figuur 4.13.

Eindige automaten

De analogie met de digitale circuitsynthese gaat nog verder en leidt tot de modellering volgens eindige automaten. Bij elke iteratie gaat het systeem over op een toestand met verdubbelde resolutie, waarbij de opeenvolgende bits van de pixelindices de toestandstransities beschrijven [181]. Deze aanpak lijkt heel sterk op de OBDD-gebaseerde technieken maar kent een aantal varianten die zich gemakkelijk laten uitbreiden naar monochrome beelden. Zelfsimilariteit en terugkerende patronen vormen de grondslag van deze aanpak. Net als bij fractale compressie kan dit voor sommige triviale beelden leiden tot een eindige maar resolutieloze beschrijving van het beeld.

Eigenschappen

Al deze technieken hebben een aantal fundamentele eigenschappen gemeen. Een inherente beperking van deze methoden is dat ze incompatibele topologieën proberen op elkaar af te beelden. De planaire structuur van een beeld kan nooit perfect afgebeeld worden op de hiërarische indexering die nodig is voor een Boolese interpretatie. Graycodering biedt hier enige hulp omdat deze lokaal het nabuurschap kan bewaren, maar globaal blijf deze inherente beperking overeind. Als gevolg hiervan is er een grote gevoeligheid aan spatiale translaties. Bovendien moeten de terugkerende patronen een dyadische aflijning hebben opdat de datastructuur ze zou kunnen incorporeren. Enkel de Boolese minimalisatie volgens een meerniveaurepresentatie kan deze beperking omzeilen. Deze laatste aanpak is in theorie optimaal, helaas is deze optimalisatie in de praktijk niet duidelijk omschreven en ook niet haalbaar.

Anderzijds zijn deze methoden gemakkelijk uit te breiden naar niet-binaire beelden, bewegende beelden en meerdimensionale volumes. Dit kan door de adressering uit te breiden aan de ingangskant van de Boolese functie en door

de uitgangsfunctie meer verfijnd te specificeren. Deze laatste kan een groter aantal uitkomsten vooropstellen of kan de grijswaarden opsplitsen volgens hun bitvlakken, eventueel na een graycodering. Het kader waarbinnen deze technieken voorgesteld zijn is dan ook heel universeel. Het concept van “don’t care”-bits, veelvuldig gebruikt in de schakelalgebra, leent zich hier uitstekend om verlieshebbende technieken op te stellen.

In een aantal situaties komen daar nog bijkomende voordelen bij. Zo kan voor een eventuele beeldverwerkingsstap de geoptimaliseerde voorstelling gebruikt worden in plaats van de oorspronkelijke voorstelling, bijvoorbeeld indien het beeld spatiaal heel groot maar ook heel ijl is. Bovendien is het mogelijk een aantal beeldbewerkingen zoals bijvoorbeeld orthogonale rotatie en spiegeling rechtstreeks op de datastructuur uit te voeren.

Wegens hun binaire karakter kunnen deze technieken niet efficiënt overweg met ruis. Ook de terugkerende patronen in tekst zijn moeilijk te onderscheppen. Omdat de meeste compressietechnieken precies uit tekstcompressie ontstaan zijn heeft deze categorie tot op heden weinig aandacht gekregen.

4.4 Methoden voor monochrome beelden

Qua computervoorstelling onderscheiden monochrome beelden zich van binaire beelden doordat de pixelintensiteit meer dan twee waarden kan aannemen. In tegenstelling tot kleurenbeelden blijft de pixelintensiteit een eendimensionale grootheid. Het is mogelijk de technieken voor binaire beelden rechtstreeks toe te passen op monochrome beelden, maar de bekomen efficiëntie is eerder beperkt. Zo kan JBIG1 monochrome beelden comprimeren door achtereenvolgens graycodering toe te passen op de pixelwaarden en contextmodellering op de resulterende bitvlakken.

Historisch zijn er dan ook technieken ontwikkeld specifiek voor monochrome beelden. De verschillen tussen de statistische eigenschappen van binaire en monochrome beelden zijn immers te groot en de methoden zijn onvoldoende universeel om zonder aanpassing ingezet te worden. De oorsprong van deze statistische verschillen is tweeledig. Veelal stellen beide beeldtypes andere informatie voor (tekstbeelden en lijnwerk versus contone beelden van natuurlijke scènes) en indien ze dezelfde informatie voorstellen, is er een fundamenteel verschil qua representatievorm (halftoonbeelden versus contone beelden).

Bovendien zijn de meeste technieken voor binaire beelden verliesloos, terwijl monochrome beelden meestal verlieshebbend gecomprimeerd worden. De belangrijkste redenen hiervoor zijn dat er slechts een beperkte compressieverhouding bereikt wordt voor verliesloze compressie van natuurlijke scènes, dat

de granulariteit van de voorstelling voldoende klein is zodat minimale wijzigingen niet te detecteren zijn en dat de generatie van een monochroom beeld inherent een verlieshebbend proces is. Toch zijn er domeinen waar verlieshebbende compressie niet wenselijk of zelfs niet toegelaten is en deze toepassingen rechtvaardigen de verliesloze beeldcompressietechnieken. Enkele voorbeelden van deze domeinen zijn medische beeldvorming, de drukvoorbereidingsindustrie, beeldvorming vanuit de ruimte via satelliet en digitale archivering van kunstwerken.

Hierna bespreken we enkele verliesloze beeldcompressietechnieken. De opsplitsing van deze technieken gebeurt volgens de impact van de transformatie (uit het hierboven beschreven paradigma) op de structuur van het beeld. Nagenoeg alle technieken gebruiken één of andere transformatie om de redundantie tussen de pixels te verkleinen. Een notoire uitzondering hierop is GIF (Graphics Interchange Format). Deze techniek is ontworpen voor paletbeelden en is louter woordenboekgebaseerd [164]. Verder merken we op dat de beschrijving van de technieken doorgaans opgaat voor alle pixels behalve deze die aan de randen van het beeld gelegen zijn. Omdat deze minder buren hebben, moeten de algoritmen hiervoor aangepast worden. De impact hiervan op de efficiëntie is evenwel te verwaarlozen.

Structuurbewarende methoden hebben een transformatiestap die de redundantie uit de voorstelling verwijdert maar de mathematische beeldstructuur ongewijzigd laat. Zowel voor als na de transformatie kan de beeldinformatie voorgesteld worden als een matrix van gehele getallen. Daarentegen vormen multiresolutietechnieken en blokgebaseerde technieken voorbeelden van technieken die de beeldstructuur ingrijpend veranderen. Multiresolutietechnieken zetten de tweedimensionale matrix van beeldelementen om in een vooraf vastgelegde piramidale structuur. Blokgebaseerde methoden splitsen het beeld op in blokken van constante of variabele grootte en vorm. Structurele technieken transformeren het beeld in een minimale variabele datastructuur.

Het is niet haalbaar om een volledig en gedetailleerd overzicht te geven van alle gepubliceerde methoden voor verliesloze compressie van beelden. We beperken ons tot de technieken die historisch een mijlpaal zijn geweest. Een conceptuele vergelijking van de belangrijkste technieken kan eveneens gevonden worden in [153, 154] en meer recentelijk ook in [32].

4.4.1 Sequentiële methoden

De structuurbewarende technieken zijn historisch de oudste en conceptueel de eenvoudigste. Ze transformeren het oorspronkelijk beeld in een gedecorreleerd beeld met dezelfde afmetingen. Hiervoor gebruiken ze *voorspelling* (*E.* pre-

diction) en om die reden worden ze ook voorspellingsgebaseerde technieken genoemd. Deze voorspelling kan variëren van de waarde van een dichtste buur, over lineaire voorspelling, niet-lineaire voorspelling, contextgebaseerde voorspelling naar voorspelling gebaseerd op neurale netwerken [200]. Omdat het oorspronkelijke beeld veel redundantie bevat, zal de entropie van de onafhankelijk en identiek gedistribueerd veronderstelde pixels in het gedecorreleerd beeld lager zijn. De compressie wordt bijgevolg bekomen door de spatiale redundantie uit te buiten. Een methode is *sequentieel* (*E.* sequential) indien ze structuurbewarend is, de rasterscanvolgorde gebruikt en daarenboven het beeld in één beweging verwerkt.

De transformatie wordt bekomen door een causale *voorspeller* (*E.* predictor) toe te passen op de pixelwaarden. Vanaf hier hanteren we expliciet een tweedimensionale indexering voor de beeldelementen. De grootheid x_{ij} stelt de pixelwaarde voor op rij j en kolom i, met $1 \leq i \leq m$ en $1 \leq j \leq n$. De voorspeller u maakt een schatting $\hat{x}_{ij}$ voor de pixelwaarde x_{ij} op basis van de pixelwaarden in een omgeving L_{ij}. De causaliteitsvoorwaarde legt aan de posities in L_{ij} de volgende voorwaarde op: na toepassing van de globale sorteerfunctie moeten alle posities uit L tot het verleden behoren. De verschillen $d_{ij} = x_{ij} - \hat{x}_{ij}$ vormen samen het *verschilbeeld* of *residubeeld* (*E.* residual image). De pixelwaarden van het verschilbeeld kunnen onafhankelijk van elkaar gecodeerd worden, of er kan nog een extra modelleerstap toegevoegd worden. De hiernavolgende technieken onderscheiden zich op het vlak van de voorspeller (lineair of niet-lineair), de eventueel aanwezige additionele contextmodellering, de entropiecoder en mogelijks aanwezige extra functionaliteit. Methoden die een classificatie uitvoeren in de toekomst, zoals woordenboekgebaseerde methoden die uit de LZ-algoritmen gegroeid zijn, komen niet aan bod [194, 234].

Sunset

De techniek *Sunset* is vooral historisch van belang omdat het de eerste verliesloze compressietechniek is voor monochrome beelden die gebaseerd is op de conceptuele scheiding tussen modelleren en coderen [133, 244]. Deze methode is opgebouwd uit de volgende drie componenten.

- **voorspelling**: Toepassing van de voorspeller u op de causale omgeving L_{ij} van x_{ij} bepaalt de voorspelling $\hat{x}_{ij}$ en de voorspellingsfout $d_{ij} = x_{ij} - \hat{x}_{ij}$.
- **contextbepaling**: De contextfunctie beeldt het verleden af op een context y_{ij} gebaseerd op een causaal sjabloon, dat kan verschillen van de causale omgeving voor de voorspelling.

- **waarschijnlijkheidsmodel**: Dit model bepaalt een schatting voor de waarschijnlijkheid $p(x_{ij}|y_{ij})$ op basis van waargenomen grootheden.

In het geval van Sunset zijn de contexten gebaseerd op voorspellingsfouten. Vele andere technieken hebben later deze aanpak overgenomen. Sunset werd zelf ook nog aanzienlijk uitgebreid en verbeterd en de naam "Sunset" verwijst vaak naar deze nieuwere geoptimaliseerde varianten [134, 135].

LJPEG

De welgekende referentiemethode *Lossless JPEG* of LJPEG is de verliesloze modus van de JPEG-standaard [119, 172, 263]. De standaard is uitgebracht door het JPEG-comité (Joint Photographic Experts Group) als gevolg van een samenwerking tussen ISO, IEC en ITU-T.

JPEG is vooral bekend als standaard voor verlieshebbende compressie van contone beelden. Hierbij wordt het beeld in 8×8-blokken opgedeeld en wordt de *discrete cosinustransformatie* (*E.* discrete cosine transform) of DCT toegepast op elk van deze blokken. Hierbij is het mogelijk een kwaliteitsparameter in te stellen die het ingevoerde verlies bepaalt door het resultaat van de transformatie op een gepaste manier te kwantiseren. Maar zelfs bij de hoogste instelling van deze parameter werkt het algoritme niet verliesloos omdat er afrondingsfouten optreden. Bovendien is deze methode minder efficiënt voor nagenoeg verliesloze compressie. De verliesloze modus, LJPEG, is minder gekend en staat volledig los van de verlieshebbende modus. Het algoritme bouwt verder op de principes van Sunset.

De modellering van LJPEG maakt gebruik van *voorspelling* (*E.* prediction) en heeft hiervoor acht verschillende lineaire voorspellers ter beschikking. Tabel 4.4 vat deze voorspellers samen waarbij "$a \div b$" de gehele deling voorstelt. Eén van deze voorspellers is de constante voorspeller (u_0), drie ervan zijn eendimensionaal (u_1, u_2 en u_3) en de overige vier zijn tweedimensionaal (u_4, u_5, u_6 en u_7). De constante voorspeller u_0 is enkel van toepassing in de hiërarchische mode, maar deze wordt hier buiten beschouwing gelaten. Tenslotte wordt het residu d berekend en gecodeerd aan de hand van een statische gewijzigde huffmancode of een aritmetische code. De LJPEG-versie die gebruik maakt van aritmetische codering voert nog een extra modelleerstap uit die gebaseerd is op Sunset. De voorspellingsfouten worden via een escapemechanisme geconditioneerd op een beperkt aantal contexten.

Deze methode is conceptueel zeer eenvoudig en bovendien kan, afhankelijk van de ingestelde parameters, de compressie en de decompressie heel snel verlopen.

Tabel 4.4: De acht voorspellers van LJPEG. Hierbij stelt '$\div$' de gehele deling voor en is $a = x_{i-1,j}$, $b = x_{i,j-1}$ en $c = x_{i-1,j-1}$.

voorspeller	functie	voorspeller	functie
u_0	0	u_4	$a + b - c$
u_1	a	u_5	$a + (b - c) \div 2$
u_2	b	u_6	$b + (a - c) \div 2$
u_3	c	u_7	$(a + b) \div 2$

FELICS

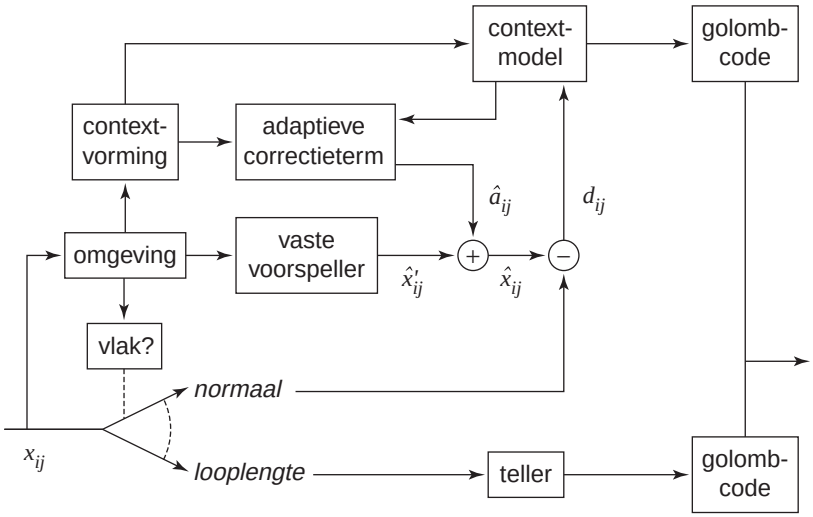
Een andere snelle en eenvoudige techniek is *Fast Efficient Lossless Image Coding System* of FELICS [97, 103]. Dit is de verliesloze compressiemethode voor monochrome beelden die deel uitmaakt van het "Managing Gigabytes" project [273]. De methode voert geen expliciete transformatie uit maar genereert impliciet een nieuwe code voor iedere pixel gebaseerd op de waarden van twee naburige pixels.

Voor iedere pixel x_{ij} op positie (i, j) wordt van de twee dichtste buren het minimum $l = \min\{x_{i-1,j}, x_{i,j-1}\}$ en het maximum $h = \max\{x_{i-1,j}, x_{i,j-1}\}$ bepaald. Vervolgens wordt de pixelwaarde x_{ij} gecodeerd aan de hand van een code gebaseerd op de waarden van l en h:

- $x_{ij} < l$: het codewoord $C(x_{ij})$ is van de vorm '$10C_r^k(l - x_{ij} - 1)$', waarbij $C_r^k(.)$ de ricecode voorstelt met parameter k;
- $l \leq x_{ij} \leq h$: de pixelwaarde x_{ij} wordt gecodeerd als '$0C_p^{h-l+1}(x_{ij} - l)$', waarbij $C_p^q(.)$ een geleidelijke binaire code voor q waarden is die de korte codewoorden toekent aan de centraal gelegen waarden;
- $h < x_{ij}$: het codewoord $C(x_{ij})$ is van de vorm '$11C_r^k(x_{ij} - h - 1)$', waarbij andermaal $C_r^k(.)$ de ricecode voorstelt met parameter k.

Deze code, die eigenlijk een samenstelling is van andere codes, wordt soms de *felicscode* genoemd. Ze is gebaseerd op de veronderstelling dat de verdeling binnen het interval $[l, h]$ vrijwel vlak is, terwijl ze daarbuiten exponentieel vervalt. De code binnen het interval is een geleidelijke code omdat de lengte van het centrale interval niet altijd een macht van 2 is.

Merk op dat de ricecode met parameter k identiek is aan de golombcode met parameter 2^k. De parameter k wordt geoptimaliseerd op basis van de waargenomen distributie. Concreet gebeurt door in een tabel voor elke combinatie



Figuur 4.15: JPEG-LS: vereenvoudigd logisch blokschema.

van $h - l$ en $k \in \{0, 1, 2, 3\}$ de voorlopige lengte bij te houden die zou bekomen zijn en telkens die waarde van k te kiezen waarvoor de voorlopige lengte minimaal wordt.

LOCO-I/JPEG-LS

Het meer recente *Low Complexity Lossless Compression for Images* of kortweg LOCO-I is net als Sunset en LJPEG een methode die vertrekt vanuit het perspectief van de scheiding van modellering en codering [266]. De techniek heeft een lage complexiteit op het vlak van benodigde rekenkracht maar is door zijn geavanceerde modellering toch heel efficiënt. Ze ligt dan ook aan de grondslag van JPEG-LS, de nieuwe ISO-ITU-standaard voor verliesloze en quasi-verliesloze compressie van contone beelden [111, 267]. Figuur 4.15 geeft het blokschema weer van JPEG-LS.

De modelleerstap combineert voorspelling met contextmodellering van het residu. De voorspeller omvat een vaste voorspeller en een adaptieve correctieterm. Als vaste voorspelling $\hat{x}'_{ij}$ voor x_{ij} wordt de mediaan van $x_{i-1,j}$, $x_{i,j-1}$ en $x_{i-1,j} + x_{i,j-1} - x_{i-1,j-1}$ genomen. Deze voorspelling wordt *Median Edge Detector* of MED genoemd. De adaptieve correctieterm $\hat{a}_{ij}$ volgt uit het contextmodel en wordt later behandeld. Het is eenvoudig na te gaan dat de vaste voorspeller niet-lineair is en een randdetecterend karakter heeft. De context van een pixel is gebaseerd op drie gekwantiseerde gradiënten binnen het cau-

saal sjabloon bestaande uit de vier meest nabije pixels. Op deze manier worden hogere-orde-afhankelijkheden geïncorporeerd in de modelvorming. De contextfunctie is gebaseerd op gradiënten en dit is efficiënter dan de contextfunctie gebaseerd op voorspellingsfouten zoals die bijvoorbeeld in Sunset gebruikt wordt. Het aantal contexten wordt nagenoeg gehalveerd door contexten met tegengestelde tekens samen te voegen en het teken van het residu desgewenst te veranderen, een techniek die gekend staat onder de naam *tekenomkering* (*E.* sign flipping). De aanpak veronderstelt verder dat het voorlopige residu $d'_{ij} = x_{ij} - \hat{x}'_{ij}$, voor een gegeven context, parametrisch kan gemodelleerd worden aan de hand van een tweezijdige geometrische distributie van de vorm $\theta^{|d'-\mu|}$. Deze distributie kent twee contextafhankelijke parameters: de centrale waarde μ en een basiswaarde θ. De centrale waarde μ kan geïnterpreteerd worden als een contextafhankelijke driftterm en bevat een geheel gedeelte $\lfloor\mu\rfloor$ en fractionele verschuiving $\sigma = \mu - \lfloor\mu\rfloor$. Op dit moment treedt de adaptieve correctieterm in actie door een schatting $\hat{a}_{ij}$ te maken voor $\lfloor\mu\rfloor$ aan de hand van waarnemingen binnen dezelfde context. Hieruit volgt tenslotte de finale voorspelling $\hat{x}_{ij} = \hat{x}'_{ij} + \hat{a}_{ij}$ en het finale residu $d_{ij} = x_{ij} - \hat{x}_{ij}$, dat verdeeld is volgens de tweezijdige geometrische distributie $\theta^{|d-\sigma|}$, met $0 \leq \sigma < 1$.

De kracht van deze aanpak volgt uit de relatie tussen de veronderstelde distributie en de efficiënte golombcode. Golombcodes zijn immers optimaal voor eenzijdige geometrische distributies van de vorm $(1-\theta)\theta^x$. De *rice-afbeelding* herordent de tweezijdige rij indices $\ldots, -2, -1, 0, 1, 2, \ldots$ in de eenzijdige rij $0, -1, 1, -2, 2, \ldots$ en beeldt op deze manier de tweezijdige distributie af op een eenzijdige. De tweedimensionale parameterruimte (θ, σ) wordt gepartitioneerd in gebieden waarbinnen een zelfde grondtal b optimaal is. Om snelheidsredenen worden enkel ricecodes gebruikt, dit zijn golombcodes waarbij het grondtal b van de vorm 2^k is. Het grondtal 2^k wordt geschat zodat het de gemiddelde absolute fout binnen een gegeven context benadert. Dit verloopt zeer efficiënt en wordt voor elke waarde opnieuw uitgevoerd.

De golombcode is veel sneller dan een aritmetische code, maar is minder optimaal. De golombcode produceert immers een codewoord voor elke bronwaarde en is daarom een karaktergebaseerde code. Bijgevolg kan de gemiddelde codewoordlengte nooit kleiner worden dan één bit. Om deze reden wordt looplengtecodering toegepast in vlakke gebieden. Telkens de gradiënten uit de context allemaal nul zijn, schakelt het algoritme over naar looplengtecodering ("run mode") en worden geen contexten meer berekend zolang alle pixels een identieke waarde hebben. In JPEG-LS wordt de looplengte gecodeerd aan de hand van een variant op een golombcode, eveneens met een adaptieve parameter (de "block-MEL-code"). Wanneer een pixel met een verschillende waarde gedetecteerd wordt, schakelt de techniek terug naar normale codering ("regu-

lar mode") en wordt de contextmodellering opnieuw geactiveerd. Deze aanpak kan dan ook beschouwd worden als een vorm van alfabetuitbreiding. Om de niet-stationariteit van beelden aan te pakken worden cumulatieve grootheden op een regelmatige basis gehalveerd. Tenslotte kan JPEG-LS ook als quasi-verliesloze techniek gebruikt worden door de voorspellingsfouten op een gepaste manier te kwantiseren.

De verschillen tussen LOCO-I en JPEG-LS zijn klein en betreffen vooral de codering van de looplengtes. De grote kracht van deze methodes zit hem in de subtiele afstemming tussen modelleer- en codeerstap. De eenvoud en efficiëntie van golombcodes wordt gecombineerd met contextmodellen.

LOCO-A is een variant van LOCO-I die gebruik maakt van aritmetische codering [267]. Meerdere contexten worden samengevoegd in contextklassen op basis van een activiteitsniveau. Deze grootheid is een lokale schatting van de variatie van de fout, meer bepaald de gemiddelde absolute fout horend bij een bepaalde context. Op deze manier wordt afgestapt van de veronderstelling van een tweezijdige geometrische distributie. Het aantal contexten voor voorspelling is groot (365 contexten in JPEG-LS), maar het aantal contextklassen voor de eigenlijke codering is klein (12 klassen in JPEG-LS). Dit is een veelgebruikte manier om het probleem van contextverdunning aan te pakken zonder over te gaan op parametrische distributies. LOCO-A is voorgesteld als een uitbreiding van JPEG-LS, namelijk "JPEG-LS Part II" [109].

LOCO-I is een voor de praktijk aangepaste projectie van de principes van *Universal Context Modeling* of UCM, dat één van de meest universele sequentiële technieken is voor compressie van monochrome beelden [264]. Deze methode is op zijn beurt een aanpassing van het algoritme "Context" dat ontworpen was voor binaire beelden [186, 191]. Het universele van de methode uit zich in een minimum aantal inperkingen van de vrijheidsgraden van de modelvorming van het beeld. Als voorspeller wordt een lineair autoregressief model gebruikt waarvan de coëfficiënten adaptief herschat worden op basis van een kleinstekwadratencriterium. Vervolgens worden de contexten van variabele lengte georganiseerd in een contextboom die voortdurend aangepast wordt. Voor iedere pixelwaarde wordt als context een knooppunt binnen deze boom gekozen dat optimaal is volgens het concept van de stochastische complexiteit. Vervolgens wordt de bijhorende laplaciaanse distributie geschat. De contexten zijn gebaseerd op de residu's binnen een causaal sjabloon, gekwantiseerd volgens het criterium van minimale stochastische complexiteit. Deze contextboom betekent dat niet alleen de waarde van de parameters adaptief is, maar ook de structuur en het aantal. De aanpak is minder universeel dan oorspronkelijk de bedoeling was. Het blijkt immers onvermijdelijk een aanzienlijk aantal vrijheidsgraden in te perken en aldus de klasse van typische beelden te

verkleinen om de convergentiesnelheid te laten toenemen. Zo gebruikt deze universele methode voorspelling, al zou dit eigenlijk niet nodig zijn voor een volkomen universele aanpak.

Merk op dat dezelfde causale omgeving van een pixel twee keer gebruikt wordt in de modelvorming: eenmaal voor de voorspelling en eenmaal voor de contextmodellering. Deze redundante aanpak kan verder geanalyseerd en geoptimaliseerd worden in de achtergrond van de modelkost [265]. Als resultaat van deze analyse volgt de aanbeveling die een "groot" contextmodel vooropstelt voor de adaptieve voorspelling en een "klein" contextmodel voor de adaptieve codering. Dit resultaat wordt bepaald door het feit dat een contextmodel voor voorspelling weinig last heeft van contextverdunning, dit in tegenstelling tot een contextmodel voor codering.

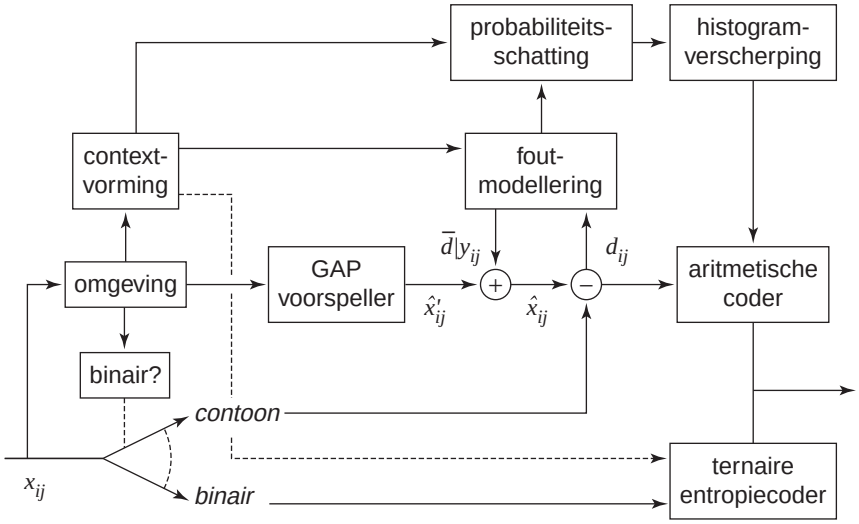
PPPM

Net als PPM maakt *Prediction by Partial Precision Matching* of PPPM gebruik van contexten van variabele lengte, maar dan voor beelden in plaats van teksten [102]. Andere bouwstenen zijn voorspelling, probabiliteiten gebaseerd op de laplacedistributie en aritmetische codering.

De techniek verwerkt de pixels volgens de rasterscan. Voor iedere pixel x_{ij} op positie (i, j) wordt een schatting $\hat{x}_{ij} = (x_{i,j-1} + x_{i-1,j} + 1) \div 2$ gemaakt en hieruit volgt het residu $d_{ij} = x_{ij} - \hat{x}_{ij}$. De context y_{ij} bestaat uit de aaneenrijging van de q meest significante bits van $x_{i-1,j}$, $x_{i,j-1}$, $x_{i-1,j-1}$ en $x_{i+1,j-1}$. Hierbij is q zo groot mogelijk, maar wel zodanig dat de resulterende context al vaker moet zijn voorgekomen dan een bepaalde drempelwaarde. Daar waar PPMB een drempelwaarde van 2 gebruikte, moet deze nu minstens 10 tot 15 zijn. Van alle residu's d waargenomen bij deze context wordt het steekproefgemiddelde $\bar{d}$ en de steekproefvariantie s_d^2 berekend. Deze variantie wordt vervolgens afgerond tot één van 37 voorgedefinieerde waarden. Vooraf is voor elk van deze voorgedefinieerde varianties een probabiliteitsdistributiefunctie berekend waarbij de laplacedistributie verondersteld wordt. Tenslotte wordt het probabiliteitsinterval horende bij de waarneming $d_{ij} - \bar{d}$ en overeenkomstig de veronderstelde probabiliteitsdistributiefunctie doorgestuurd naar een aritmetische coder.

CALIC

De gouden standaard in verliesloze compressie van monochrome beelden wordt gevormd door *Context-based, Adaptive, Lossless Image Codec* of CALIC. De methode is gebaseerd op niet-lineaire voorspelling, tweevoudige contextmodellering en aritmetische codering [278, 283]. Op een hoger algorit-



Figuur 4.16: Het vereenvoudigd logisch blokschema van CALIC. Om de analogie met JPEG-LS duidelijk te maken, wijkt deze voorstelling af van de gangbare voorstelling in [283].

misch niveau vertoont ze opmerkelijk veel gelijkenissen met JPEG-LS. In verhouding tot andere algoritmen is ze vrij rekenintensief maar ze behaalt de beste compressieresultaten van alle gekende technieken en is zelfs beter dan UCM. De oorspronkelijke versie van CALIC werd ingediend als voorstel voor de nieuwe verliesloze JPEG-standaard en overleeft daar als een alternatief dat tragere maar betere compressie biedt dan JPEG-LS. De methode is gebaseerd op eerder onderzoek naar contextclassificatie voor beeldcompressie en werd kort na indienen nog lichtjes verbeterd [279, 280].

Figuur 4.16 vat de werking van CALIC samen in een vereenvoudigd blokschema. Het algoritme kent twee modi: een contone modus voor gebieden met natuurlijke variatie zoals die meestal voorkomt in digitale beelden van natuurlijke scènes en een binaire modus voor gebieden die synthetisch gegenereerd zijn en slechts één of twee pixelwaarden hebben. Het aantal verschillende pixelwaarden in een causale omgeving van zeven pixels bepaalt de modus waarin de pixel gecodeerd wordt. Indien hoogstens twee verschillende waarden optreden, wordt de binaire modus gebruikt, anders de contone modus. De binaire modus maakt dan ook gebruik van een ternaire aritmetische coder die, naast de twee pixelwaarden uit de omgeving, toelaat om andere pixelwaarden te coderen aan de hand van een escapemechanisme.

De contone modus vormt het hart van het algoritme en verloopt onafhan-

kelijk van de binaire modus. Deze modus bestaat uit drie grote componenten: voorspelling, contextmodellering van de voorspellingsfouten en tenslotte entropiecodering. Op zijn beurt bestaat de voorspelling uit twee subcomponenten. Ten eerste is er een vaste *gradient-adjusted prediction* voorspeller of GAP-voorspeller. Deze statische voorspeller gebruikt een vrij complexe niet-lineaire functie die de zeven causale pixelwaarden uit de omgeving van pixel (i, j) afbeeldt op een voorspelling $\hat{x}'_{ij}$. Daartoe worden de horizontale en verticale gradiënt in de omgeving bepaald. Op basis van deze gradiënten wordt één voorspeller gekozen uit zes beschikbare lineaire voorspellers. Ten tweede is er een adaptieve correctieterm die gebaseerd is op een contextafhankelijke terugkoppeling van het voorlopige residu $d'_{ij} = x_{ij} - \hat{x}'_{ij}$. Deze term laat toe om texturen van hogere-orde-structuren en lokale activiteit gezamenlijk te modelleren. Het resultaat $\hat{x}_{ij}$ van de voorspelling bestaat uit de som van de vaste term $\hat{x}'_{ij}$ en het contextgebaseerde gemiddelde residu $\bar{d}|y_{ij}$, waarbij y_{ij} de context van pixel (i, j) voorstelt. De contextfunctie die beschrijft hoe y_{ij} volgt uit de waarde van de omgeving laat 576 verschillende contexten toe. Deze contextafbeelding incorporeert tegelijkertijd verschillende eigenschappen zoals spatiale texturen enerzijds en de grootte van de residu's anderzijds.

Omdat deze voorspelling de statistische redundantie in het beeld niet volledig elimininеert, wordt contextmodellering nog eens gebruikt om het finale residu $d_{ij} = x_{ij} - \hat{x}_{ij}$ te modelleren. Om contextverdunning te vermijden is het aantal contextklassen deze keer gereduceerd tot 8. Elk van deze 8 contextklassen beschrijft de grootte van de pixelvariatie in de omgeving en kan dan ook geïnterpreteerd worden als een energieschatting. Ook hier wordt gebruik gemaakt van tekenomkering. Indien de omgeving wijst op een negatief teruggekoppeld residu $\bar{d}|y_{ij}$, dan wordt niet het residu d_{ij} maar het tegengesteld residu $-d_{ij}$ gecodeerd. Dit heeft een verscherping van het conditioneel histogram als gevolg. Tenslotte drijft dit tweede contextmodel een aritmetsiche coder aan. Om nog meer adaptiviteit toe te laten, worden de statistieken op regelmatige basis herschaald.

De gelijkenissen tussen JPEG-LS en CALIC zijn treffend. Beide maken gebruik van een binaire en een contone modus, van een vaste niet-lineaire voorspelling, van een adaptieve correctieterm gebaseerd op contextmodellering en van contextmodellering van het residu op basis van een beperkt aantal contexten. Ze onderscheiden zich in de details van de voorspelling, de contextfunctie, het aantal contexten, het type entropiecoder en een groot aantal implementatiedetails.

Het is evenwel paradoxaal dat CALIC betere resultaten haalt dan UCM. Deze laatste is ontworpen om een theoretische ondergrens te bepalen voor wat mogelijk is en is, veel meer dan CALIC, gebaseerd op theoretische principes.

De verklaring is dat UCM helemaal niet zo optimaal is als het zelf beweert: de halfoneindige contextruimte moet als een tweedimensionale entiteit behandeld worden en mag niet zonder meer afgebeeld worden op een eendimensionale rij, het causale sjabloon voor voorspelling is kleiner dan dat van CALIC, de convergentie verloopt trager door een groter aantal vrijheidsgraden en lineaire autoregressie mag dan optimaal zijn voor de grootte van de voorspellingsfouten, het is niet optimaal voor de entropie ervan.

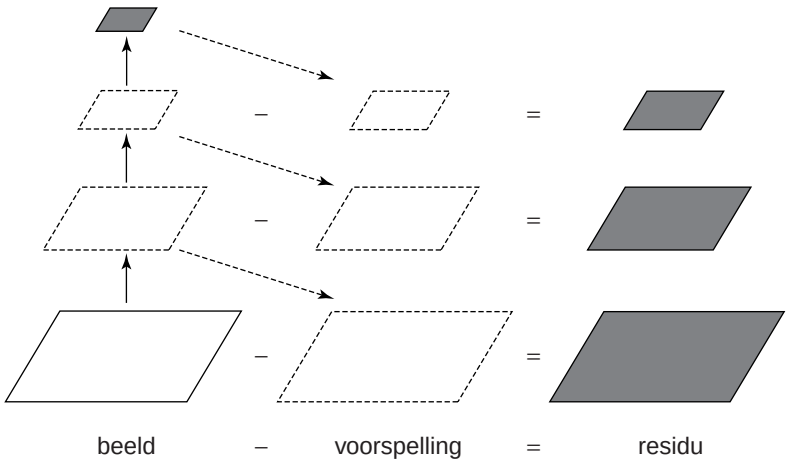
4.4.2 Multiresolutietechnieken

De zopas beschreven sequentiële methoden hebben veel aantrekkelijke eigenschappen maar hebben één groot gebrek: de ingebouwde data-afhankelijkheid maakt het onmogelijk om een beeld progressief te decoderen of om een arbitrair stuk van een beeld te decoderen zonder daarvoor eerst de causale omgeving te reconstrueren. Deze eigenschappen zijn in de praktijk nochtans heel nuttig en worden aangeboden door multiresolutietechnieken. Deze technieken transformeren het vlakke beeld in een hiërarchische structuur van een beperkt aantal lagen waarbij elke hogere laag beeldinformatie voorstelt op een lagere resolutie. Merk op dat het associëren van hogere lagen aan een lagere resolutie tegengesteld is aan de situatie bij JBIG.

De *resolutiereductiefunctie* zet een beeld van hoge resolutie om in een beeld van lagere resolutie. Vertrekkend van een origineel beeld wordt deze functie iteratief een aantal keren toegepast. Onderbemonstering vormt de kern van de resolutiereductiefunctie. Hieruit resulteert een piramidale structuur. De reconstructie volgt het omgekeerde pad.

Er wordt echter pas compressie bekomen indien de hogeresolutiebeelden efficiënt kunnen voorgesteld worden. Omdat een voorspelling kan gemaakt worden gebaseerd op het voorlopige lageresolutiebeeld, volstaat het verschilbeeld tussen voorspelling en hogeresolutiebeeld als additionele informatie. Meestal neemt die voorspelling de vorm aan van *interpolatie*. Net als bij sequentiële methoden wordt de term *residu* gebruikt voor het verschilbeeld tussen de voorspelling en het eigenlijke hogeresolutiebeeld. Voor natuurlijke beelden kan het equivalente verschilbeeld efficiënter voorgesteld worden dan het hogeresolutiebeeld.

De resolutiereductie wordt slechts een beperkt aantal keer uitgevoerd zodat de bekomen hiërarchische structuur als een *afgeknotte piramide* (*E.* truncated pyramid) kan voorgesteld worden. De toplaag van de piramide stelt het laagsteresolutiebeeld voor en van de andere lagen zijn enkel de verschilbeelden van belang. Tot slot kan het laagsteresolutiebeeld gecomprimeerd worden met een structuurbewarende methode. Figuur 4.17 stelt de hiërarchische ontbinding



Figuur 4.17: Hiërarchische ontbinding van een beeld. De opwaartse volle pijlen en de schuine gestreepte pijlen stellen respectievelijk de resolutiereductiefunctie en de voorspelling voor. Het originele beeld staat linksonder en de resulterende piramide wordt voorgesteld door de grijze vlakken.

van een beeld voor. Voor de verschillagen geldt duidelijk de relatie "beeld − voorspelling = residu" en bijgevolg ook "beeld = voorspelling + residu".

Progressieve reconstructie wordt meestal bekomen door de lagen door te sturen van laagste tot hoogste resolutie. Dit staat gekend onder de term "progressiviteit volgens resolutie". Er is evenwel nog een andere manier om progressieve reconstructie te bekomen. Bij "progressiviteit volgens kwaliteit" worden achtereenvolgens de bitvlakken dwars over de lagen heen doorgestuurd, van meest significant naar minst significant bitvlak. Hiervoor wordt ook wel de term "progressiviteit volgens pixeldiepte" gebruikt. Indien contextmodellering gebruikt wordt, heeft het type progressiviteit een grote invloed op de optimale sjabloonkeuze.

Het reduceren van de resolutie kan tweevoudig of viervoudig zijn: hierbij stemt één pixel uit het lageresolutiebeeld overeen met respectievelijk twee en vier pixels uit het hogeresolutiebeeld. De hiermee corresponderende datastructuur is respectievelijk een *tweeboom* of *binaire boom* (*E.* binary tree) en een *vierboom* (*E.* quadtree). Merk op dat deze begrippen hier een andere betekenis krijgen dan bij de structurele technieken voor binaire beelden.

Het totaal aantal waarden neemt toe van mn in het oorspronkelijke beeld naar $mn(1-1/\alpha)^{-1}(1-(1/\alpha)^k)$ in de boomstructuur, met $\alpha = 2$ en $\alpha = 4$ voor respectievelijk tweebomen en vierbomen en k het totaal aantal lagen. Voor $k \to \infty$ wordt dit respectievelijk $2\,mn$ en $4/3\,mn$. Maar deze toename kan

vermeden worden door een goede keuze te maken voor de resolutiereductiefunctie en de voorspelling. Het gebruik van decimatie als onderbemonstering, waarbij bepaalde pixelwaarden onveranderd gekopieerd worden en andere niet in rekening gebracht worden, is een duidelijk voorbeeld hiervan. Nadeel van decimatie is dat het *frequentieverwarring* (*E.* alias) met zich meebrengt en dit komt de kwaliteit van het beeld op lage resolutie niet ten goede.

De inherente kracht van multiresolutietechnieken situeert zich in het opheffen van de vereiste dat de pixels sequentieel verwerkt moeten worden. Het is dan ook toegelaten om voorspellings- en contextsjablonen te gebruiken die niet causaal zijn volgens de rasterscan. Dit argument heeft ook een keerzijde: terwijl voor bepaalde pixels een sjabloon met grotere voorspellende kracht gebruikt kan worden, zal voor andere pixels een sjabloon met een veel kleinere voorspellende kracht moeten gebruikt worden.

Multiresolutietechnieken onderscheiden zich van elkaar op de volgende vlakken: boomtype, resolutiereductiefunctie, voorspelling tussen de lagen, compressiemethode voor de toplaag, statistisch model voor de voorspellingsfouten, globale sorteerfunctie bepalend voor het type progressiviteit en entropiecoder. Oorspronkelijk waren deze technieken meer populair in het verlieshebbend domein dan in het verliesloos domein omdat de optie van progressiviteit haaks staat op de vereiste van verliesloze reconstructie. Sommige technieken slagen erin om de voordelen van goede verlieshebbende en goede verliesloze compressie te integreren in één algoritme.

HINT

De techniek *Hierarchical INTerpolation* of HINT is historisch de oudste multiresolutietechniek voor verliesloze compressie van beelden [72, 202]. De techniek maakt gebruik van een tweeboom bestaande uit vijf lagen. Als resolutiereductiefunctie wordt decimatie genomen zodat er geen toename is in aantal pixelwaarden. De voorspelling is een eenvoudige lineaire interpolatie van de dichtste buren. De toplaag wordt gecomprimeerd aan de hand van een sequentiële methode gebaseerd op voorspelling. De verschillagen worden gecomprimeerd aan de hand van een statische entropiecode. Het compressievermogen is beperkt omdat de toplaag en alle verschillagen onafhankelijk van elkaar behandeld worden. De methode is progressief volgens resolutie en de kwaliteit van het laagsteresolutiebeeld vertoont vaak artefacten als gevolg van de decimatie.

MLP

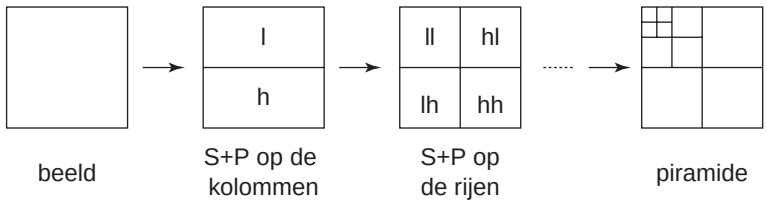
Multi-Level Progressive of MLP is een rekenintensieve multiresolutietechniek gebaseerd op HINT [100, 101]. Ook hier wordt gebruik gemaakt van tweebomen gebaseerd op decimatie. De lineaire voorspelling van de tussenlagen maakt gebruik van de meest nabijgelegen 16 causale buren. De piramide hoeft niet langer afgeknot te zijn maar kan eindigen in een toplaag van vier pixels. De voorspellingsfouten worden gemodelleerd volgens een laplaciaanse distributie met centrale waarde 0 en variantie σ^2. Deze distributies zijn voorafgaand uitgerekend en getabelleerd. De methode stelt meerdere manieren voor om deze variantie te schatten en vervolgens te kwantiseren. Tenslotte geeft een tabel het corresponderende waarschijnlijkheidsinterval terug dat op zijn beurt naar de aritmetische coder wordt doorgestuurd. Ook hier is de visuele kwaliteit van de lageresolutiebeelden laag.

BTPC

Een minder rekenintensieve variant is *Binary Tree Predictive Coding* of BTPC [195]. Deze is eveneens gebaseerd op decimatie en maakt gebruik van binaire bomen met acht lagen. De toegevoegde waarde van deze techniek zit in de niet-lineaire voorspelling en de aangepaste codering. De niet-lineaire voorspelling is gebaseerd op de vorm van het viertal opgebouwd uit de pixelwaarden van de vier dichtste buren binnen de hogere laag. Hun relatieve grootte wordt geclassificeerd in veertien onderscheiden situaties en voor elk van deze situaties wordt een voorspellingsfunctie gedefinieerd die steunt op het principe van de “dichtste tegenovergestelde buren”. Dit principe kan beschouwd worden als een uitbreiding van de definitie van de mediaan naar een situatie waar er vier waarden gegeven zijn. Als entropiecode worden huffmancodes gebruikt, aangevuld met een speciaal *nulboomsymbool* dat aangeeft dat alle waarden onder een bepaalde node in de tweeboom nul zijn. De techniek kan tenslotte verlieshebbend gemaakt worden door de voorspellingsfouten te kwantiseren.

S+P

Het gebruik van decimatie als onderbemonstering heeft een negatief effect op de beeldkwaliteit van de voorlopige reconstructie en op het compressievermogen. De methode *Sequential and Predictive* of S+P gebruikt een betere onderbemonstering dan louter decimatie en combineert die met een extra voorspelling om zowel goede verliesloze als goede verlieshebbende compressie te bekomen [205, 206]. Maar bij een dergelijke onderbemonstering moet voorzichtig tewerk gegaan worden om een mogelijke data-expansie te vermijden.



Figuur 4.18: Constructie van een afgeknotte piramide door herhaaldelijke toepassing van de S+P-transformatie op achtereenvolgens kolommen en rijen. Enkel de rechthoek van de vorm "ll...l" bevat een lageresolutiebeeld terwijl alle andere waarden residuele informatie voorstellen.

De hier gebruikte transformatie slaagt daar uitstekend in en kan beschouwd worden als een verliesloze vorm van de *wavelettransformatie*. Deze zijn op hun beurt gebaseerd op *subbandcodering* [277]. Goede verlieshebbende reconstructie wordt eenvoudig bekomen door de gecomprimeerde stroom voortijdig af te breken.

De S+P-transformatie bestaat uit de samenstelling van vooreerst de S-transformatie die de onderbemonstering en een eerste voorspelling uitvoert en vervolgens de P-transformatie die een tweede voorspelling uitvoert binnen het verschilbeeld. De gezamenlijke transformatie is gedefinieerd in één dimensie en wordt achtereenvolgens op de kolommen en de rijen uitgevoerd. Figuur 4.18 illustreert hoe het herhaaldelijk toepassen van de S+P-transformatie aanleiding geeft tot een afgeknotte vierboom. Uit de figuur blijkt ook dat er geen toename qua aantal pixelwaarden optreedt.

In één dimensie beschouwd is de S-transformatie ("sequential transform") gelijkaardig aan de *haartransformatie*. Ze zet een rij pixelwaarden x_i met $i = 1, \ldots, n$ en n even om in een laagfrequente rij $l_j = (x_{2j-1} + x_{2j}) \div 2$ en een hoogfrequente rij $h_j = x_{2j-1} - x_{2j}$ met $j = 1, \ldots, n/2$. Het is eenvoudig aan te tonen dat deze transformatie reversibel is. De inverse transformatie genereert $x_{2j-1} = l_j + (h_j + 1) \div 2$ en $x_{2j} = x_{2j-1} - h_j$. Het aantal waarden neemt niet toe, al verdubbelt het dynamisch bereik van de hoogfrequente rij h_j en gebruikt de voorstelling van de waarden na toepassen van de S-transformatie bijgevolg één extra bit.

Voor beeldcompressie garandeert deze S-transformatie een betere progressieve beeldkwaliteit dan onderbemonstering gebaseerd op decimatie. Maar omwille van het slechte frequentieantwoord van de filteroperatie is er nog altijd een vrij grote redundantie in de hoogfrequente rij. De P-transformatie heeft tot doel dit frequentieantwoord te verbeteren door een voorspelling $\hat{h}_j$ te maken voor de waarden h_j uit de hoogfrequente rij. De voorspelling $\hat{h}_j$ is een

lineaire combinatie van l_k met $k = 1, \ldots, n/2$ en h_k met $k = j+1, \ldots, n/2$. De P-transformatie ("predictive transform") vervangt vervolgens h_j door het verschil $h_j - \lfloor \hat{h}_j + 1/2 \rfloor$. Als filter worden er drie vaste lineaire combinaties voorgesteld die tot doel hebben het antwoord in het frequentiedomein te verbeteren. Met elk van deze drie filters stemt een andere schatting van de frequentieinhoud overeen.

De techniek kan kiezen uit meerdere entropiecoders. Naast de klassieke huffmancode, is er ook een variant van aritmetische codering en een nieuwe methode gebaseerd op nulbomen. De variant van aritmetische codering maakt gebruik van contexten en classificeert eerst de fout volgens grootte. Vervolgens wordt de fout gecodeerd door aangepaste aritmetische codering van de foutklasse en ongecodeerde transmissie van het teken en de grootte van de fout binnen de klasse. De codering op basis van *nulbomen* (*E.* zerotrees) is vrij nieuw en is gebaseerd op verlieshebbende compressietechnieken die gebruik maken van wavelets zoals *Set Partitioning In Hierarchical Trees* of SPIHT [207] en *Embedded Zerotree Wavelet* of EZW [224]. Deze zogenaamde *nulboomcodering* (*E.* zero tree coding) stuurt bitvlak na bitvlak door en is bijgevolg progressief volgens kwaliteit. De structuur binnen een bitvlak, eveneens een boomstructuur, wordt gerespecteerd en de term *nulboom* stelt een tak voor die enkel uit '0'-bits bestaat. Deze wordt voorgesteld door het speciale nulboomsymbool. Nulboomcodering is geen vorm van entropiecodering, omdat ze niet expliciet op statistieken is gebaseerd. De resulterende stroom is bijgevolg doorgaans niet optimaal, maar de codeer- en decodeerstap verloopt zeer snel.

Eén van de belangrijkste eigenschappen van de gecomprimeerde stroom is dat hij nagenoeg volledig *ingebed* (*E.* embedded) is. Dit betekent dat de beeldkwaliteit van de reconstructie gebaseerd op een prefix van de gecomprimeerde stroom een optimum bereikt voor de kwaliteit die bekomen kan worden door een verlieshebbende coder die een gecomprimeerde stroom van dezelfde lengte genereert. Dit is helemaal niet het geval bij structuurbewarende methoden.

JPEG2000

Parallel aan de verliesloze compressiestandaard JPEG-LS, groeide de nood aan een nieuwe verlieshebbende compressiestandaard. De huidige verlieshebbende standaard JPEG, gebaseerd op de DCT, ontbeert namelijk een aantal essentiële eigenschappen voor de technologieën en toepassingen van de nieuwste generatie. De hoogste prioriteit bij het opstellen van JPEG2000 was gericht op het verhogen van de functionaliteit en het uitbreiden van het toepassingsdomein, eerder dan de efficiëntie of de snelheid te verbeteren [121]. Het is niet

de bedoeling dat deze nieuwe standaard JPEG zal vervangen.

Het toepassingsgebied van deze nieuwe standaard strekt zich uit van printen, scannen en het Internet over kleurenfax, digitale fotografie en mobiele beeldvorming tot remote sensing, medische beeldvorming en digitale beeldarchieven. Dit geeft aanleiding tot beelden met heel verschillende karakteristieken en de modellering vereist dan ook meer dan één enkele klasse van beeldmodellen. In vergelijking met JPEG staan de volgende eigenschappen en beperkingen aan de basis van deze nieuwe standaard [89, 211, 229]:

- **schaalbaarheid**: Prioritair is dat de beeldkwaliteit bij hoge compressieverhoudingen beter wordt en dat de artefacten minder storend zijn. Aan de andere kant van het compressiespectrum resulteert het finaal stadium van progressieve verlieshebbende compressie in verliesloze reconstructie. De dynamische pixeldiepte kan variëren van 1 bpp voor binaire beelden tot 16 bpp per component voor contone beelden met hoge tonale resolutie. De methode is toepasbaar op zowel heel kleine beelden als heel grote beelden. Er zijn voorzieningen voor kleurenbeelden en meercomponentenbeelden.

- **data-afhankelijkheid**: De reconstructie kan progressief zijn zowel volgens resolutie als volgens pixeldiepte, dit om de reconstructie optimaal aan te passen aan de eigenschappen van het uitvoerapparaat. Er is willekeurige toegang (*E.* random access) tot het beeld, wat ook bewerkingen zoals translatie en rotatie toelaat in het gecodeerde domein. Prioritaire reconstructie van een statisch of dynamisch *interessegebied* (*E.* region of interest) of ROI is mogelijk indien dit gebied bij codering gespecificeerd wordt. Er is voorzien in de mogelijkheid tot sequentiële codering en decodering in één beweging, wat waretijdscodering mogelijk maakt. De afhankelijkheid van voorgaande beeldlijnen is beperkt, zodat ook de vertraging en de geheugenvereisten beperkt blijven.

- **robuustheid**: De gecomprimeerde stroom is beter bestand tegen bitfouten, dit vooral met het oog op draadloze transmissie. De situatie waar één bitfout een volledig beeld kan verstoren is hierdoor vermeden. De volgende aspecten zijn van belang: datapartitionering, resynchronisatie, foutdetectie, foutverbetering en transmissie gebaseerd op *servicekwaliteit* (*E.* quality of service) of QoS. Schaalbaarheid laat toe om robuustheid eenvoudig te implementeren.

- **architectuur**: Een open architectuur voorziet in een *kern* (*E.* core) aanwezig in elke decoder en *uitbreidingen* (*E.* extensions) die op aanvraag aangeroepen en opgestuurd worden.

Deze doelstellingen worden bereikt door drie fundamentele wijzigingen ten opzichte van JPEG. Ten eerste worden de beelden opgesplitst in tegels en deze worden volkomen onafhankelijk van elkaar behandeld. Ten tweede is de decorrelatie niet langer gebaseerd op de DCT maar op de wavelettransformatie. Ten derde is het gecomprimeerd beeldformaat efficiënt georganiseerd en bovendien uitbreidbaar, wat toelaat om het beeldformaat aan te passen aan specifieke behoeften.

Tegels Vooreerst wordt het beeld opgesplitst in componenten, bijvoorbeeld kleurcomponenten, en hierbij zijn twee componenttransformaties gespecificeerd binnen de standaard. Deze zijn respectievelijk irreversibel en reversibel en ze stellen het beeld voor in een andere kleurenruimte die ofwel een betere compressie toelaat, ofwel dichter aanleunt bij het HVS. Vervolgens splitst het algoritme elke component op in *tegels* (*E.* tiles), wat vooral de geheugenvereisten en de beeldtoegankelijkheid ten goede komt. De tegels zijn rechthoeken van willekeurige maar constante afmetingen. Alvorens de tegel te transformeren, wordt het dynamisch bereik van de component bepaald. De helft van het dynamisch bereik wordt afgetrokken van elke pixelwaarde van de tegel, een stap die gekend staat als *DC-verschuiving* (*E.* DC level shifting). Als resultaat hiervan zijn de pixelintensiteiten niet langer tekenloos. Daarna wordt de DWT toegepast op elke tegelcomponent.

Transformatie De *discrete wavelettransformatie* (*E.* discrete wavelet transform) of DWT die instaat voor de decorrelatie van de data vertoont gelijkenissen met HINT en S+P. De wavelettheorie genereert een mathematisch kader dat toelaat een multiresolutieanalyse uit te voeren op (meer)dimensionale signalen [259]. Toepassing van een *analysefilter* op een eendimensionaal signaal ontbindt dit signaal in enerzijds een lageresolutiesignaal met gehalveerde resolutie en anderzijds een onderbemonsterd hogeresolutiesignaal dat als residu fungeert. Het *synthesefilter* volgt uit het analysefilter en voert de omgekeerde operatie uit.

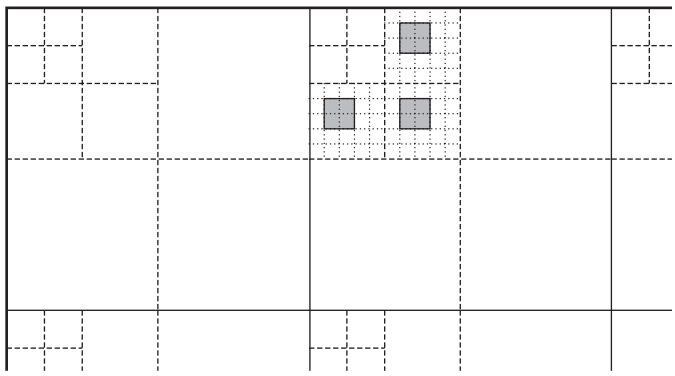
De JPEG2000-standaard specificeert twee standaardfilters voor de wavelettransformatie, maar laat daarnaast ook toe om zelf een filter op te geven. Het Daubechies $(9, 7)$-standaardfilter biedt een hogere beeldkwaliteit bij verlieshebbende reconstructie, maar is irreversibel en laat bijgevolg niet toe om verliesloze compressie uit te voeren. Het Le Gall $(5, 3)$-filter daarentegen is wel reversibel en laat zowel verlieshebbende als verliesloze reconstructie toe. De notatie "(k, l)" betekent hierbij dat het analysefilter k en l filtercoëfficiënten heeft voor respectievelijk het laag- en het hoogdoorlaatfilter. De implementatie van een filter kent twee varianten. Voor beiden wordt het signaal eerst *even-*

periodisch (*E.* periodic symmetric) uitgebreid ter hoogte van de randen. De eerste implementatievariant is gebaseerd op convolutie en wordt gebruikt voor het irreversibele Daubechies $(9, 7)$-filter. De tweede implementatievariant is gebaseerd op een *heffingsschema* (*E.* lifting scheme). Deze bestaat uit een opeenvolging van eenvoudige filteroperaties die alternerend de oneven waarden aanpassen met een gewogen som van even waarden en de even waarden aanpassen met een gewogen som van de oneven waarden. Een implementatie gebaseerd op een heffingsschema verloopt efficiënter dan een rechtstreekse implementatie van convolutie. Het heffingsschema wordt bij voorkeur toegepast voor het reversibel Le Gall $(5, 3)$-filter.

Kwantisatie en codering In het geval van irreversibele compressie worden de waveletcoëfficiënten vervolgens scalair gekwantiseerd. Deze bewerking is verantwoordelijk voor nagenoeg het volledige verlies aan beeldkwaliteit (daarnaast is er nog het minieme verlies door afrondingsfouten). De kwantisatiestap is verschillend voor elke subband en is gebaseerd op het dynamisch bereik binnen elke subband.

Tenslotte worden de waarden gecodeerd aan de hand van contextmodellering en een variant van aritmetische codering. Net als JBIG2 wordt hiervoor gebruik gemaakt van de MQ-coder maar het aantal contexten is beperkt tot 9. Zoals geïllustreerd in figuur 4.19, wordt de getransformeerde data nog verder opgesplitst. Elke subband wordt opgesplitst in *codeblokken* (*E.* code-blocks), die onafhankelijk van elkaar gecodeerd worden. Deze aanpak is gebaseerd op *Embedded Block Coding with Optimized Truncation* of EBCOT [238]. De bitvlakken binnen een codeblok worden één na één doorgestuurd en elk bitvlak binnen een codeblok vergt drie bewegingen [142]. Een aantal codeblokken overeenkomend met eenzelfde spatiale rechthoek in drie subbanden van eenzelfde resolutie wordt gebundeld in een *precinct* (*E.* precinct). Een *pakket* (*E.* packet) bevat een aantal gecodeerde bitvlakken van de codeblokken horend bij eenzelfde precinct. Een verfijnd bit-allocatie-algoritme beperkt de grootte van de pakketten en organiseert deze in *kwaliteitslagen* (*E.* layers). Het aantal kwaliteitslagen kan sterk variëren en is bepalend voor het type progressiviteit.

Het opsplitsen in codeblokken, pakketten en kwaliteitslagen is heel flexibel en brengt een aantal interessante eigenschappen met zich mee. De resulterende codestroom is *parseerbaar* (*E.* parseable), wat betekent dat pakketten kunnen gedecodeerd worden zonder de voorgaande te decoderen. Als gevolg hiervan kan de stroom progressief gemaakt worden volgens resolutie, kwaliteitslaag, positie of component. Bovendien laat deze aanpak willekeurige toegang toe, beperkt het de geheugenvereisten voor zowel software- als hardware-implementaties, en vereenvoudigt het het ontwerp van een parallelle



Figuur 4.19: JPEG2000 deelt de getransformeerde data op in kleine blokken. Het beeld (vette lijn, "—") wordt opgesplitst in tegels (volle lijn, "—") waarop een 3-niveau-DWT wordt toegepast. Elke subband (streeplijn, "- - -") bevat een aantal codeblokken (stippellijn, "····"). De figuur toont één precinct dat 12 codeblokken groepeert (grijze rechthoeken).

implementatie.

Complexiteit De toename aan functionaliteit en de vooropgestelde eisen van schaalbaarheid en data-afhankelijkheid zijn bereikt, maar dit gebeurt niet zonder enige kost. De implementatie op zich is vrij complex en om die reden zal er een referentieimplementatie beschikbaar gesteld worden. De huidige implementaties van JPEG2000 zijn grofweg driemaal trager dan de geoptimaliseerde JPEG-implementaties en ook de geheugenvereisten zijn beduidend toegenomen. Het blijft nog wachten op snellere implementaties, maar nu al staat vast dat JPEG2000 niet in de buurt zal komen van JPEG. De berekeningscomplexiteit van de wavelettransformatie in combinatie met de bitgeoriënteerde aanpak van de aritmetische coder kan niet wedijveren met de combinatie van de DCT en huffmancodering. Daartegenover staat de mogelijkheid tot parallelliseren en de keuzevrijheid van tegelgrootte en codeblokken wat toelaat om de geheugenvereisten te beperken.

JPEG2000 slaagt er in een groot bereik aan interessante eigenschappen te bundelen in een eenvormig kader. De standaard zal bestaan uit meerdere delen. Deel I (*E.* Part I) beschrijft de minimale voorwaarden waaraan een compatibele decoder moet voldoen en deze zijn allemaal publiek beschikbaar en vrij van licenties. Deel II (*E.* Part II) betreft uitbreidingen voor specifieke toepassingen en meer complexe optimalisaties. Daarna volgen nog een aantal delen, zoals bijvoorbeeld Deel III over "Motion JPEG2000" voor het editeren

van videobeelden aan hoge kwaliteit.

De aanzet tot het ontwikkelen van deze nieuwe standaard werd ten dele gegeven door *Compression with Reversible Embedded Wavelets* of CREW [19, 286]. Dit voorstel was kandidaat voor de JPEG-LS standaard maar heeft het daar moeten afleggen tegen LOCO-I. Andere technieken die mee bepalend zijn geweest voor de finale vorm zijn S+P, EZW, SPIHT, *Wavelet/Trellis Coded Quantization* of WTCQ en EBCOT [206, 207, 220, 224, 238].

4.4.3 Blokgebaseerde en andere technieken

Voor verliesloze compressie zijn de sequentiële technieken doorgaans de meest efficiënte omdat ze de redundantie in het beeld maximaal uitbuiten. Helaas heeft het sequentieel karakter een negatieve impact op de data-afhankelijkheid. Multiresolutietechnieken hebben dit probleem niet en bieden meer functionaliteit zonder veel aan efficiëntie in te boeten. Daarnaast zijn er nog een heel aantal technieken met een transformatie die niet in beide voorgaande categorieën thuis te brengen zijn. Deze technieken zijn meestal oorspronkelijk ontworpen voor verlieshebbende compressie en later aangepast voor verliesloze compressie.

Technieken voor vaste blokken

Blokgebaseerde methoden splitsen het beeld op in een groot aantal kleine blokken die onafhankelijk van elkaar behandeld worden. Bij blokgebaseerde methoden met vaste blokken zijn deze blokken van dezelfde grootte en vorm en zijn het meestal vierkanten. Hier moet enkel de afmetingen van één blok doorgestuurd worden als parameter. Bij blokgebaseerde methoden met variabele blokken kan de grootte en de vorm verschillen van blok tot blok. In dit geval is de structuur van het getransformeerde beeld beeldafhankelijk en dus moet zowel de vorm als de positie van de blokken doorgestuurd worden als parameters. Daarnaast treedt er in het codeerproces een extra stap op die verantwoordelijk is voor het eigenlijk opsplitsen van het beeld. Deze stap wordt de *segmentatie* genoemd en hieruit volgt de term *segmentatiegebaseerde* technieken.

Ongetwijfeld de meest populaire blokgebaseerde techniek met vaste blokken is JPEG. Hierbij wordt de DCT toegepast op blokken van grootte 8×8, gevolgd door huffmancodering. De DCT maakt gebruik van vlottendekommagetallen en is om die reden inherent verlieshebbend. De BinDCT is een numerieke benadering van de DCT die gehele getallen afbeeldt op gehele getallen en bijgevolg verliesloze compressie toelaat [138, 245]. De constructie van de BinDCT is gebaseerd op het *heffingsschema* (*E.* lifting scheme) dat toelaat om snelle verliesloze transformaties te construeren [236]. De efficiëntie van deze

aanpak kan niet wedijveren met sequentiële methoden. De kracht schuilt zich evenwel in het verenigen van verlieshebbende met verliesloze compressie en de inherente snelheid. Niet alleen gebeuren alle berekeningen op gehele getallen, maar daarenboven is de implementatie vrij van vermenigvuldigingen.

Ook andere transformaties kunnen aangepast worden zodat ze voor verliesloze beeldcompressie in aanmerking komen. De techniek *Lossless Hadamard* of LHAD is gebaseerd op de *hadamardtransformatie* toegepast op blokken van 8×8 pixels. De methode past de LDU-decompositie toe op de hadamardmatrix om een efficiënte kwantisatieprocedure op te stellen zonder de reversibiliteit in gedrang te brengen [177].

Segmentatiegebaseerde technieken

Segmentatiegebaseerde technieken worden gekenmerkt door een extra bewerking die enkel tijdens het codeerproces optreedt: het opsplitsen van het beeld in segmenten. Deze segmentatie is cruciaal voor zowel de efficiëntie als de snelheid tijdens het coderen. De segmenten worden doorgaans gedefinieerd als gebieden waarbinnen de beeldintensiteit weinig varieert. De gecomprimeerde stroom bestaat uit de lijst van de segmenten met hun posities en vorm, evenals de textuur binnen de segmenten. Deze technieken ontleden het beeld in zijn opbouwende elementen en gaan hierdoor verder dan de voorgaande technieken die het beeld beschouwen als een matrix van toevalsgrootheden. Om deze reden behoort deze klasse van technieken tot de compressiemethoden van de tweede generatie.

Segmentation-based Lossless Image Coding of SLIC is een segmentatiegebaseerde techniek die specifiek ontworpen is voor medische beelden [225]. De techniek segmenteert het beeld in gebieden met min of meer constante beeldintensiteit. Het resultaat hiervan evenals het resulterende foutbeeld worden gecodeerd aan de hand van JBIG.

Ruimtevullende curves

Sequentiële technieken maken gebruik van de rasterscan als globale sorteerfunctie. Deze is eenvoudig en ligt intuïtief voor de hand maar heeft minder goede lokaliteitseigenschappen. De causale buurt van een pixel is immers nooit groter dan het bovenliggende halfvlak. In principe kan elke *ruimtevullende* (*E.* space-filling) curve gebruikt worden als sorteerfunctie. Een curve is ruimtevullend indien ze elk punt van de ruimte (of het beeld) precies één keer passeert. De *hilbertscan* en de *peanoscan* zijn beide ruimtevullende curves die interessante lokaliteitseigenschappen hebben [204]. Ook de globale sorteerfunctie van multiresolutietechnieken bepaalt een ruimtevullende curve. Elke

ruimtevullende curve heeft omwille van de gewijzigde causaliteit een niet-triviaal effect op zowel de voorspelling als op de contextmodellering.

Bij deze minder voor de hand liggende ruimtevullende curves is de causale buurt voor een aantal pixels kleiner dan bij de rasterscan, maar voor andere pixels is ze veel groter. Het is dan ook de vraag of ze in staat zijn betere compressieresultaten op te leveren. Experimenteel onderzoek heeft uitgewezen dat er weinig of geen winst mag verwacht worden. Dit is in lijn met eigen onderzoeksresultaten. Theoretisch onderzoek op contextmodellering bij een klasse van abstracte beeldmodellen wijst uit dat de winst door de contexten van de meer causale sjablonen teniet gedaan wordt door het verlies van de minder dekkende sjablonen [152].

De hierboven vermelde ruimtevullende curves zijn statisch in de zin dat ze beeldonafhankelijk zijn. Indien we nog een stap verder gaan, dan kunnen dynamische ruimtevullende curves overwogen worden die geoptimaliseerd zijn voor elk beeld. Wordt een beeld beschouwd als een volledig geconnecteerde graaf, dan definieert elke opspannende boom een ruimtevullende structuur of een *scanmodel*. Dit scanmodel kan niet langer als een eendimensionale curve voorgesteld worden maar wel als een boomstructuur. Een optimaal scanmodel miminaliseert de totale absolute voorspellingsfout en laat bijgevolg toe deze efficiënt te coderen. Helaas is de kost van het doorsturen van dit optimale scanmodel heel groot. Het verkleinen van de keuzevrijheid en het opstellen van een codeboek van scanmodellen kan een oplossing bieden [155].

Structurele technieken

De tot hiertoe besproken methoden passen een transformatie toe op een beeld en maken gebruik van een entropiecoder om de gedecorreleerde datastructuur te coderen. De structuur na transformatie of na toepassen van de globale sorteerfunctie ligt nagenoeg vast en de beeldinformatie is aanwezig in de waarden die de datastructuur aanneemt.

Een aantal methoden zijn geïnspireerd op de structurele technieken voor binaire beelden. Ze buiten de redundantie in het beeld uit door er een geoptimaliseerde datastructuur uit af te leiden. Deze datastructuur is dan ook volledig afhankelijk van het te coderen beeld. Een bekende techniek is *Weighted Finite Automata* of WFA, waarbij het beeld geïdentificeerd wordt met een eindige automaat [43]. De vier toestandstransities vertrekkend vanuit een vierkant blok stemmen overeen met de intensiteitsovergangen binnen elk van de vier subkwadranten en worden beschreven aan de hand van een lineaire combinatie van de intensiteiten van de bestaande toestanden. Elke transitie wordt gekenmerkt door gewichten die het aantal van de samenstellende toestanden beschrijven.

De aanpak is leerrijk voor beelden die een hoge mate aan zelfsimilariteit hebben maar faalt op realistische beelden.

4.5 Methoden voor meercomponentenbeelden

Bij meercomponentenbeelden stelt iedere pixel een vector voor in een kleurenruimte. Het beeld bevat meestal drie of vier componenten en de meest voorkomende kleurenruimtes zijn RGB, YUV, LAB, YIQ en CMYK. In een aantal situaties zoals bij satellietbeelden kan het beeld bestaan uit veel meer componenten.

Een triviale benadering zou zijn om elke component apart te comprimeren door er een methode voor monochrome beelden op toe te passen. Maar op deze manier wordt de *tonale* correlatie, dit is de correlatie tussen de componenten, niet in rekening gebracht. Een aantal technieken en standaarden houden hier wel expliciet rekening mee. We maken een onderscheid of ze al dan niet de kleurenruimte als gekend beschouwen.

4.5.1 Gekende kleurenruimte

JPEG-LS uitbreiding

JPEG-LS specificeert naast een splitsing van de componenten ook een *alternerend* (*E*. interleaved) formaat voor kleurenbeelden [111]. Hierbij worden de componenten alternerend per pixel of per lijn doorgestuurd. De standaard ondersteunt geen expliciete kleurtransformaties, en zowel de voorspelling als de contextvorming vinden volledig plaats binnen eenzelfde component. De tonale decorrelatie reikt niet verder dan het gemeenschappelijk stellen van de statistieken en de modus (looplengtecodering versus normale codering). Kleurtransformaties en expliciete tonale decorrelatie worden er verwezen naar de applicatielaag. De auteurs van LOCO-I beschrijven nochtans hoe voor een aantal kleurenruimtes een significante winst kan bekomen worden met een eenvoudige statische kleurtransformatie. Als voorbeeld beschrijven ze de RGB-transformatie die een (r, g, b) kleurenvector afbeeldt op $(r - g, g, b - g)$ met een aangepaste modulo-verschuiving [267]. Deze aanpak gaat enkel op voor RGB-beelden en is dan ook niet universeel over alle kleurenruimtes heen of robuust tegen afwijkende tonale statistieken.

JPEG2000

JPEG2000 is de enige standaard die officieel kleurtransformaties ondersteunt [121]. Deel I definieert twee statische transformaties die enkel opgaan

voor RGB-beelden. De *Reversible Component Transformation* of RCT is reversibel en komt zowel voor verliesloze als voor verlieshebbende compressie in aanmerking. Ze beeldt een (r, g, b)-triplet af op $((r+2g+b) \div 4, b-g, r-g)$, waarbij de laatste twee waarden een verdubbeld dynamisch bereik vertonen. Daarnaast is er ook de *Irreversible Component Transformation* of ICT die enkel kan gebruikt worden voor verlieshebbende compressie. Deel II bevat de nodige uitbreidingen om adaptieve kleurtransformaties te ondersteunen. Deze zijn bijvoorbeeld gebaseerd op de *Karhunen-Loève-transformatie* of KLT die een optimale decorrelatie uitvoert vanuit een energetisch perspectief.

4.5.2 Ongekende kleurenruimte

IB-CALIC

Zoals de naam laat vermoeden bouwt *Interband CALIC* of IB-CALIC verder op het succes van CALIC [282, 284]. De techniek is verregaand geoptimaliseerd en behaalt keer op keer een signifcante winst in efficiëntie door de tonale correlatie uit te buiten. De componenten worden geordend volgens een niet nader omschreven techniek en één na één verwerkt. Elke vorige component dient als *referentieband* (*E.* reference band) voor de huidige component. De lokale correlatie tussen de componenten wordt geschat en indien die te laag is, valt de techniek terug op het klassieke gedrag van CALIC. De causale buurt in de referentieband is eveneens beperkt tot een halfvlak. Strikt genomen zou die de volledige referentieband mogen bedekken, maar dan wordt de contextgeneratie afhankelijk van het type alternering (pixel, lijn of component). De omgeving in de referentieband wordt gebruikt voor zowel de voorspelling als voor de contextgeneratie. Er zijn verregaande voorzieningen om de berekeningscomplexiteit te drukken, zowel voor de schatting van de correlatie, voor het berekenen van de voorspelling, als voor het bepalen van de context. Het criterium om over te schakelen naar binaire modus wordt uitgebreid naar de referentieband.

SICLIC

De *Simple Inter-Color Lossless Image Coder* of SICLIC is een eenvoudige uitbreiding van LOCO-I naar meercomponentenbeelden [12]. In tegenstelling tot de eerder beschreven uitbreiding van JPEG-LS worden geen veronderstellingen gemaakt betreffende de kleurenruimte. Eén component, groen in het geval van RGB-beelden, wordt gekozen als referentieband en wordt gecodeerd aan de hand van LOCO-I. De andere componenten maken gebruik van deze referentieband indien het voordelig is. Net als LOCO-I kent de methode twee

soorten codering: normale codering en looplengtecodering. In normale codering zijn twee voorspellers tegelijkertijd actief, één die geen gebruikt maakt van de referentieband en één die er wel gebruik van maakt. De contexten worden gegenereerd op basis van de gekwantiseerde verschillen tussen de huidige en de referentiecomponent. De techniek gebruikt een causaal criterium voor de keuze van de voorspeller, namelijk de grootte van de contextgebaseerde gemiddelde voorspellingsfout. Voor looplengtecodering wordt een onderscheid gemaakt of de looplengtes *verbonden* (*E.* joint) zijn of niet, wat betekent dat ze ook in de referentiecomponent optreden of enkel in de huidige component.

Verliesloze KLT

De Karhunen-Loève-transformatie of KLT, ook wel *Hotelling-transformatie* genoemd, is een transformatie die vooral voor verlieshebbende compressie gebruikt wordt. Ze is gebaseerd op een schatting van de covariantiematrix van de waargenomen data en daaruit volgt dat de transformatiematrix een data-afhankelijk karakter vertoont. De eigenlijke transformatiematrix bestaat uit de genormeerde eigenvectoren van de covariantiematrix van de componenten. Deze aanpak minimaliseert de variantie binnen de artificiële componenten na transformatie en vormt aldus een optimale decorrelatie van de kleurenruimte. Toepassen van de LDU-decompositie op de transformatiematrix in combinatie met een gepaste kwantisatie maakt de KLT ook bruikbaar voor verliesloze compressie [257]. Hierbij kan de KLT toegepast worden op blokken van vaste grootte of op segmenten van variabele vorm en grootte.

4.6 Methoden voor meerdimensionale beelden

Meerdimensionale beelden onderscheiden zich van meercomponentenbeelden doordat elk tweedimensionaal beeld een lichtjes gewijzigde scène voorstelt. De derde dimensie is dan ook een ruimtelijke dimensie en niet langer een spectrale dimensie. Het ontstaansproces van driedimensionale datasets kan een belangrijke rol spelen bij de compressie ervan. Zo zijn de medische volumes niet langer fotografisch maar tomografisch van aard in twee dimensies, wat zijn gevolgen heeft op de ruiseigenschappen van de beelden. Daarenboven zijn de datasets meestal *anisotroop* omdat een eenheidsvolume niet dezelfde afmetingen heeft in elke dimensie. Merk op dat bij videobeelden de derde dimensie temporaal is. Niettegenstaande dezelfde methoden als voor meerdimensionale beelden vaak toepasselijk zijn, gaan we hier niet dieper op in.

Het toevoegen van een derde dimensie brengt weinig fundamentele vernieuwingen met zich mee. De meeste technieken voor verliesloze compres-

sie van beelden zijn gemakkelijk uitbreidbaar. Enkele methoden voor meercomponentenbeelden zijn rechtstreeks toepasbaar op volumebeelden, al verschilt de derde spatiale dimensie qua interpretatie van een tonale dimensie. Er zijn sequentiële technieken gebaseerd op lineaire en niet-lineaire voorspelling [178]. Daarnaast bestaan er ook driedimensionale uitbreidingen van een aantal multiresolutietechnieken, gebaseerd op onder andere HINT en SPIHT [127, 201]. De belangrijkste conclusie is dat de derde dimensie doorgaans wezenlijk verschilt van de eerste twee en dat dit een belangrijke impact kan hebben op de compressie, zowel in positieve als in negatieve zin.

4.7 Experimentele resultaten

De state of the art bevat een heel groot gamma aan technieken met sterk uiteenlopende eigenschappen. Elke techniek is gekenmerkt door een ander compromis inzake efficiëntie, complexiteit, universaliteit, functionaliteit, geheugenkost en rekenkost. Daarenboven zijn bepaalde implementaties verregaand geoptimaliseerd voor snelheid, terwijl andere enkel het verliesloos karakter in de praktijk willen aantonen.

Hierna volgt een kort overzicht van de efficiëntie die een aantal technieken halen op gestandaardiseerde tekstbestanden en representatieve testbeelden. Appendix A geeft een overzicht van de testbeelden en hun eigenschappen. Het is niet onze bedoeling om hier een exhaustief overzicht te geven van de resultaten van elke techniek op elk testbeeld. Voor binaire beelden drukken we de efficiëntie uit aan de hand van de compressieverhouding, voor de andere bestanden gebruiken we de empirische entropie of *bitdiepte* (*E.* bit rate) uitgedrukt in bits per pixel of bpp. We behandelen achtereenvolgens enkele resultaten voor tekstbestanden, voor binaire beelden, voor monochrome beelden en voor meercomponentenbeelden. Daarna komt de snelheid van een aantal technieken even aan bod. Resultaten op beelden met een hoge bitdiepte (10 of 12 bpp) evenals op driedimensionale beeldensets worden achterwege gelaten. Specifieke resultaten voor medische beelden zijn gepubliceerd in [61, 65, 130, 170, 201, 202].

4.7.1 Tekstbestanden

De efficiëntie van de algemene methoden wordt het best getoetst aan de hand van de testbestanden van het Calgary en het Canterbury corpus. Omdat deze corpussen ook andere bestanden bevatten dan louter tekstbestanden, vermelden we expliciet de resultaten voor “book1” van het Calgary corpus. De resultaten worden samengevat in tabel 4.5. Voor de technieken met instelbare parameters

Tabel 4.5: Compressieresultaten uitgedrukt in bpp op een Engelse tekst "book1" en gemiddelden over het Calgary en het Canterbury corpus. De instelbare parameters zijn 'q' voor de maximale orde, 'b' voor de blokgrootte en 'm' voor de geheugengrootte.

techniek	book1	Calgary	Canterbury	opmerkingen
compress	3.30	3.63	3.30	
GZIP-9	3.25	2.70	2.53	
PPMC	2.52	2.49	2.19	zie [158]
PPMC-h	2.32	2.39	2.23	$q = 4$
PPMD	2.34	2.33	2.11	$q = 5$
PPMD+	2.30	2.28	2.12	zie [239]
PPMZ	2.21	2.12	2.00	$q = 8$ (v9.1)
PPMZ2	2.20	2.09	2.01	(v0.7)
DMC	2.48	2.58	2.39	$m = 50$ Mbyte
ACB	2.32	2.20	2.10	(v2.00c)
BZIP-9	2.40	2.34	2.15	$b = 900$ Kbyte
BZIP2-9	2.42	2.37	2.23	$b = 900$ Kbyte
SZIP-B	2.35	2.33	2.16	$q = 10$, $b = 4.1$ Mbyte
CTW1	2.16	2.23	–	$q = 8$, zie [262]
CTW2	2.19	2.19	–	$q = 8$, PPMDE
VW98	2.15	2.12	–	zie [261]

worden enkel de resultaten weergegeven voor de globaal beste parametercombinatie.

De experimentele resultaten tonen duidelijk hoe statistische methoden veel hogere compressie halen dan de LZ-gebaseerde methoden. De beste PPM-technieken en CTW kunnen wedijveren met elkaar, kort daarna gevolgd door bloksortering. Het wisselschema VW98 combineert CTW met een andere techniek (LZ77 of PPMDE) en haalt het beste resultaat voor de Engelse tekst [261]. Merk op dat PPMZ en PPMZ2 geoptimaliseerd zijn naar de testcorpussen toe door een aantal heuristieken in te voeren.

4.7.2 Binaire beelden

Voor binaire beelden onderzoeken we de efficiëntie van zowel een aantal algemene technieken als van de technieken specifiek ontworpen voor binaire beelden. Er wordt een onderscheid gemaakt tussen de tekstbeelden of zakendocumenten enerzijds en zuivere halftoonbeelden van natuurlijke scènes anderzijds. Appendix A beschrijft de eigenschappen van de testbeelden en reproduceert

een aantal van hen aan lage resolutie. Namen van beelden voldoen aan de conventie "<scène>-<kleur>{-<detail>}", tenzij anders gespecificeerd. Voor een aantal veel voorkomende testbeelden zijn aliassen voorzien.

Voor de algemene technieken gelden dezelfde parameters en opmerkingen als in tabel 4.5. De techniek T-PB is gebaseerd op "PackBits", een eenvoudig compressieschema dat looplengtecodering toepast volgens de bytegrenzen. De techniek T-LZW past LZW toe op de beelddata, eveneens volgens de bytegrenzen. Beide technieken worden gebruikt in het TIFF-beeldformaat en presteren verre van optimaal voor binaire beelden. Daarna komen de ITU-aanbevelingen T.4 (G3) en T.6 (G4) aan bod. De technieken G3-1D en G3-2D stellen respectievelijk $K = 1$ en $K = 2$. De techniek G4 stelt impiciet $K = \infty$ en laat een aantal controle- en synchronisatiesignalen achterwege. De beschreven resultaten voor G3 en G4 wijken in zeer beperkte mate af van de literatuur doordat de overhead van het TIFF-formaat wordt meegerekend. Het statisch karakter van deze technieken heeft als gevolg dat de efficiëntie afhangt van de gekozen associatie tussen de kleur van de pixels (zwart en wit) en de waarde van de bits ('0' en '1') anderzijds. Waar nodig wordt de optimale associatie afgeleid.

De techniek JBIG maakt gebruik van de standaardwaarden voor de instelbare opties. Dit betekent dat er 4 lagen zijn (1 bodemlaag en 3 verschillagen) en elk beeld wordt opgeplitst in een aantal apart behandelde *stroken* (*E.* stripes). De maximale verplaatsing voor de AT-pixel bedraagt respectievelijk 8 en 0 pixels in horizontale en verticale richting. Verder zijn de drie optionele verbeteringen actief: deterministische voorspelling, typische voorspelling in de verschillaag en typische voorspelling in de bodemlaag. De techniek JBIG-S is eveneens gebaseerd op JBIG, maar is zuiver sequentieel en behandelt de data in één laag en één strook. Dezelfde beperkingen voor het AT-pixel blijven gelden en dezelfde optionele verbeteringen zijn actief. De techniek MG-2L maakt gebruik van een twee-niveau-contextmodel met het standaard 22/10-sjabloon uit figuur 4.8. De techniek TIC tenslotte is patroongebaseerd en maakt intern gebruik van MG-2L en PPMC. Beide implementaties zijn afkomstig van het "Managing Gigabytes" project [275]. Van JBIG2 zijn er helaas geen implementaties publiek beschikbaar. Tot slot is er de techniek OBDD die gebaseerd is op geordende binairebeslissingsdiagrammen [150].

CCITT testbeelden

Alle CCITT testbeelden stellen gescande zakendocumenten voor aan een resolutie van 200 dpi en hebben dezelfde afmetingen, zie figuur A.1. Tabel 4.6 stelt de compressieverhoudigen voor van een aantal technieken op elk van de testbeelden evenals het gemiddelde over de testset heen. De tabel bevat wei-

nig verrassingen. De technieken die gebruik maken van contextmodellering behalen duidelijk de beste resultaten. Hun efficiëntie is gemiddeld dubbel zo hoog als wat de beste algemene technieken halen. Het patroongebaseerde TIC scoort het best maar is dan ook de meest complexe techniek. JBIG-S haalt bijna even goede resultaten met een veel eenvoudigere en intrinsiek snellere aanpak. Het gebruik van een twee-niveau-contextmodel leidt tot betere resultaten dan JBIG-S, maar scoort minder goed dan TIC. De algemene techniek PPMC-h scoort verrassend slecht en dit is te verklaren door de beperkte precisie van de probabiliteit en het gebruik van de bereikcoder. De techniek OBDD kan niet goed overweg met ruis en schiet dan ook schromelijk tekort op deze realistische beelden.

Binnen JBIG2, dat ontbreekt in de tabel, zijn twee verliesloze aanpakken gedefinieerd. Voor de contextgebaseerde verliesloze modus, gelijkaardig aan JBIG1, zijn geen resultaten beschikbaar. De patroongebaseerde verliesloze modus is gebaseerd op SPM. Deze laatste doet het nog 10% beter dan TIC, al is dit resultaat gebaseerd op de officiële testbeelden die lichtjes afwijken van de hier gebruikte testbeelden [98].

JBIG Stockholm testbeelden

De beelden van de JBIG Stockholm testset hebben allen een resolutie van 400 dpi en kunnen verder opgesplitst worden in een aantal klassen. Het gaat om tekstbeelden, digitale halftoonbeelden, een gemengd beeld, gedigitaliseerd lijnwerk en digitaal gegenereerd lijnwerk.

Tabel 4.7 stelt de gemiddelde compressieverhouding voor op elk van deze klassen. De grotere verscheidenheid aan beelden geeft aanleiding tot een aantal verrassingen. De resultaten voor de tekstbeelden en het gedigitaliseerd lijnwerk zijn kwalitatief gelijk aan die voor de CCITT testbeelden. Maar voor de halftoonbeelden geven de statische technieken G3 en G4 aanleiding tot expansie in plaats van compresie. Zoals verwacht liggen de resultaten voor het gemengd beeld tussen de resultaten voor de tekstbeelden en de halftoonbeelden. Voor het digitaal gegenereerd lijnwerk worden uitzonderlijk hoge compressieverhoudingen behaald door de meest verfijnde modellen. De kolom met de gemiddelde efficiëntie toont aan hoe desastreus de gevolgen kunnen zijn van het gebruik van een statische methode. De expansie op enkele beelden neemt bijna alle winst weg die voor de andere beelden bekomen wordt.

Tabel 4.6: Compressieverhoudingen van de algemene en de binaire technieken voor de CCITT testbeelden "ccitt*n*-b" met $n = 1$–8 van figuur A.1. De rechterkolom geeft het gemiddelde weer.

techniek	1	2	3	4	5	6	7	8	gem.
I - Algemene technieken									
compress	15.85	18.64	8.89	4.96	8.25	12.16	4.61	10.02	8.41
GZIP-9	17.25	20.55	11.64	5.11	9.79	17.82	4.68	12.90	9.55
PPMC-h	9.01	10.54	6.27	4.19	6.04	7.64	3.77	8.50	6.23
PPMD+	19.47	22.81	10.77	5.76	10.05	14.33	5.38	12.05	10.00
BZIP2-9	18.92	23.46	12.10	5.61	10.31	19.47	5.11	14.50	10.40
SZIP	19.16	21.90	10.76	5.74	9.71	14.03	5.23	12.44	9.86
T-PB	8.83	8.31	4.84	3.06	4.70	5.34	2.27	3.99	4.31
T-LZW	14.50	17.14	8.41	4.45	7.74	11.59	4.23	9.63	7.79
II - Technieken voor binaire beelden									
G3-1D	13.63	14.84	7.86	4.74	7.49	9.99	4.81	8.14	7.67
G3-2D	17.03	20.62	10.40	5.66	9.78	14.19	5.71	11.84	9.85
G4	28.01	46.57	17.74	7.39	15.82	30.42	7.38	26.57	15.44
JBIG	30.49	57.29	21.71	8.74	18.27	38.01	8.46	33.91	18.20
JBIG-S	35.03	60.24	23.38	9.46	19.87	40.84	9.13	36.08	19.72
MG-2L	36.21	69.50	25.18	10.14	21.21	45.97	9.91	39.77	21.32
TIC	37.60	62.50	22.70	12.70	20.30	41.00	9.80	34.70	21.60
OBDD	17.41	29.71	11.80	5.10	10.43	20.14	5.35	17.53	10.51

Halftoonbeelden

Alle halftoonbeelden van de BG[1] testset stellen de cyaancomponent voor van de natuurlijke scène "musicians" gerasterd aan de hand van een hogekwaliteitsraster. Tabel A.3 beschrijft de halftoonparameters en figuur A.5 toont een uitvergroting van de testbeelden. De implementatie van PPMD+ kon niet overweg met de afmetingen van de bestanden en daarom werden enkele aanpassingen aangebracht aan de broncode. De dots zijn onvoldoende geïsoleerd voor een techniek als TIC en de resultaten ontbreken in de tabel. Voor JBIG2 is geen broncode publiek beschikbaar en ook deze resultaten ontbreken.

Tabel 4.8 vat de resultaten samen. De belangrijkste conclusie is dat de efficiëntie op halftoonbeelden aanzienlijk verschilt van die op tekstbeelden. De

[1]Deze beelden zijn ter beschikking gesteld door de firma Barco Graphics NV, recent hernoemd tot Esko-Graphics NV.

Tabel 4.7: Compressieverhoudingen van de algemene en de binaire technieken voor de JBIG Stockholm testbeelden "s*n*-b" van figuur A.2. De resultaten zijn gegroepeerd per beeldklasse. De rechterkolom geeft het gemiddelde weer.

techniek (beeld *n*)	tekst s01-3, 5	digitaal halftoon s04a-d, 9	gemengd s06	gedigit. lijnwerk s07	digitaal lijnwerk s08, 10	gem.
I - Algemene technieken						
compress	12.06	3.23	4.28	16.04	27.30	5.78
GZIP-9	12.69	3.73	4.46	30.41	44.93	6.64
PPMC-h	7.28	3.51	3.78	7.75	6.93	4.89
PPMD+	14.03	3.92	5.30	19.72	26.97	6.94
BZIP2-9	13.45	3.96	5.03	32.40	58.97	7.13
SZIP	13.81	4.00	5.30	20.31	30.45	7.05
T-PB	5.34	1.84	2.22	6.55	7.64	3.00
T-LZW	11.18	2.90	3.83	15.29	23.99	5.22
II - Technieken voor binaire beelden						
G3-1D	11.24	0.73	2.02	13.65	16.04	1.65
G3-2D	14.91	0.64	1.79	20.13	23.51	1.48
G4	24.64	0.62	1.64	46.50	53.39	1.46
JBIG	30.71	4.12	7.15	65.15	160.90	8.60
JBIG-S	31.87	4.82	6.54	69.67	204.89	9.70
MG-2L	35.83	4.85	9.14	76.67	258.80	10.21

algemene adaptieve methoden zoals SZIP en PPMD+ halen aanzienlijk betere resultaten dan de statische technieken voor binaire beelden zoals G3 en G4. Het is bovendien merkwaardig dat JBIG-S het slechter doet dan JBIG. Ook hier worden de beste resultaten behaald door MG-2L.

Dit gedrag is als volgt te verklaren. Ten eerste wijken de statistische eigenschappen van deze beelden in sterke mate af van tekstbeelden. Halftoonbeelden moeten dan ook als atypisch beschouwd worden voor technieken als G3 en G4. Ten tweede is het essentieel dat een techniek "voldoende ver" terugkijkt in het verleden om te leren uit terugkomende situaties en dat de conditionerende omgeving "voldoende groot" is. Een aantal technieken zoals JBIG-S schieten op dit vlak tekort. We komen hier in het volgende hoofdstuk uitgebreid op terug.

Tabel 4.8: Compressieverhoudingen van de algemene en de binaire technieken voor de BG halftoonbeelden uit tabel A.3. De rechterkolom geeft het gemiddelde weer.

techniek	ro5k	ro10k	ro20k	mo10k	mo20k	gem.
I - Algemene technieken						
compress	4.87	2.83	3.78	2.37	4.04	3.35
GZIP-9	4.91	3.27	3.72	2.85	15.97	4.20
PPMC-h	4.94	4.05	3.93	2.64	5.04	3.90
PPMD+	6.25	4.79	5.19	2.98	5.54	4.64
BZIP2-9	5.53	4.52	6.15	2.68	11.17	4.87
SZIP	5.96	4.78	5.16	2.87	7.09	4.72
T-PB	1.38	1.08	1.36	1.10	1.29	1.23
T-LZW	4.15	2.29	2.88	2.19	3.80	2.87
II - Technieken voor binaire beelden						
G3-1D	2.31	2.13	2.97	1.53	2.15	2.12
G3-2D	2.72	2.55	3.48	1.44	2.59	2.35
G4	3.50	3.20	5.42	1.38	4.45	2.89
JBIG	7.31	4.65	7.83	4.39	14.47	6.44
JBIG-S	5.60	4.30	7.32	4.16	13.98	5.82
MG-2L	5.99	4.94	8.75	5.88	9.96	6.63

4.7.3 Monochrome beelden

De monochrome testbeelden omvatten niet alleen beelden die de luminantie voorstellen, maar ook beelden die een zekere fysische intensiteit voorstellen (zoals de medische beelden) en de individuele componenten van kleurenbeelden.

Tabel 4.9 geeft de compressieresultaten weer van een breed gamma verliesloze technieken op een aantal testsets beschreven in appendix A. De individuele kleurcomponenten van meercomponentenbeelden worden onafhankelijk van elkaar behandeld. De technieken zijn opgesplitst in twee categorieën: algemene technieken en technieken voor monochrome beelden. De algemene technieken zijn dezelfde als deze die bij tekstbestanden en binaire beelden aan bod kwamen. De technieken voor monochrome beelden bevatten vaak een aantal instelbare opties. Hierbij wordt geopteerd voor die opties die de hoogste efficiëntie behalen zonder de implementatie te wijzigen. JBIG-S stelt sequentiële JBIG-codering voor van de minimale implementatie in combinatie met graycodering van de bitvlakken. De implementatie van LJPEG kiest de

Tabel 4.9: Compressieresultaten van de algemene technieken en de technieken voor monochrome beelden op de volgende testsets: JPEG beelden, medische beelden, RGB-beelden, JPEG-LS beelden, BG contone beelden en ISO-SCID beelden, zie appendix A. De efficiëntie is uitgedrukt als gemiddelde bitdiepte (bpp) per component.

techniek (componenten)	JPEG (9)	medisch (12)	RGB (12)	JPEG-LS (22)	BG (28)	SCID (32)	gem. (115)
I - Algemene technieken							
compress	6.62	4.54	7.12	5.31	5.56	6.24	5.84
GZIP-9	6.31	4.46	6.70	5.10	5.35	6.02	5.61
PPMC-h	5.71	3.89	6.29	4.80	4.99	5.52	5.18
PPMD+	5.39	3.63	6.01	4.46	4.63	5.22	4.86
BZIP2-9	5.08	3.44	5.57	4.25	4.40	4.99	4.61
SZIP	4.99	3.32	5.54	4.18	4.33	4.91	4.53
T-PB	7.98	6.82	8.01	6.38	6.82	7.45	7.13
T-LZW	7.47	5.08	8.04	6.09	6.29	7.17	6.65
II - Technieken voor monochrome beelden							
JBIG-S	4.54	3.49	5.29	4.16	4.48	4.77	4.49
LJPEG	4.64	3.61	5.28	4.59	4.84	4.98	4.73
FELICS	4.44	3.54	5.17	4.30	4.55	4.73	4.50
JPEG-LS	4.06	3.08	4.85	3.83	4.19	4.41	4.13
CALIC	3.92	2.99	4.73	3.71	4.06	4.28	4.00
BTPC	4.42	3.40	5.19	4.17	4.58	4.75	4.48
S+P-H	4.22	3.38	4.95	4.16	4.47	4.58	4.36
JPEG2000	4.15	3.23	4.96	3.98	4.38	4.56	4.28

beste voorspeller voor elk beeld en maakt verder gebruik van huffmancodes. FELICS bevat geen instelbare opties. JPEG-LS maakt gebruik van de standaardwaarden voor de instelbare parameters voor contextkwantisatie en voor het herstartinterval voor de statistieken (64 lijnen). LOCO-I is niet opgenomen in de tabel, maar doet het normaal net iets beter dan JPEG-LS. Voor CALIC zijn geen instelbare opties aanwezig. S+P-H maakt gebruik van huffmancodes. De waveletdecompositie van JPEG2000 is gebaseerd op het $(5, 3)$-filter en wordt 5 keer toegepast. Het beeld wordt beschouwd als één tegel, de codeblokken bevatten 64×64 pixels en er wordt geen gebruik gemaakt van precincts.

Binnen de categorie van algemene technieken halen de methoden gebaseerd op bloksortering de beste resultaten, van nabij gevolgd door de PPM-

gebaseerde techieken. Dit is omdat blokgebaseerde technieken van nature de niet-stationariteit in rekening brengen. Binnen de technieken voor monochrome beelden halen de contextgebaseerde sequentiële technieken zoals CALIC en JPEG-LS telkens de beste resultaten. CALIC haalt gemiddeld een compressieverhouding 2.0 en doet het hiermee ongeveer 3% beter dan JPEG-LS. De beste multiresolutietechnieken doen het ongeveer 8% slechter dan CALIC, met achtereenvolgens JPEG2000 (7%) en S+P-H (9%). Daarna komt de minder geavanceerde multiresolutietechniek BTPC (12%). De sequentiële technieken zonder contextmodellering halen gelijkaardige resultaten als de beste algemene technieken, waarbij FELICS (12% slechter dan CALIC) het voortdurend beter doet dan LJPEG (18%). JBIG-S tenslotte haalt betere resultaten dan LJPEG.

De medische beelden vertonen merkelijk meer redundantie dan de andere beelden. Het optreden van een egale donkere achtergrond ligt aan de grondslag hiervan. Wat RGB-beelden betreft kan gesteld worden dat het ontstaansproces niet direct aanleiding geeft tot een verschillende mate van redundantie in de respectieve componenten. Bij CMYK-beelden is de situatie verschillend. De K-component ontstaat op een fundamenteel andere manier dan de andere componenten. De apparatuur die de CMYK-beelden genereert werkt immers van nature in de RGB-kleurenruimte. De CMY-componenten volgen rechtstreeks uit deze RGB-waarden en delen dan ook de statistische eigenschappen. De K-component daarentegen wordt er artificieel aan toegevoegd omwille van een aantal druktechnische redenen zoals spectrale onzuiverheden in de inkten, kostprijs, absorptievermogen van het papier en kleureffecten. Als gevolg hiervan treden er meer egale vlakken op en is de entropie beduidend lager. Tabel 4.10 illustreert dit voor de componenten van het beeld “musicians-cmyk”, een typisch voorbeeld uit de drukvoorbereidingsindustrie. De K-component kan gemiddeld tot 30% beter gecomprimeerd worden. De grootte van het verschil in voorstelling is beeldafhankelijk, maar de tendens is consistent over alle CMYK-beelden die natuurlijke scènes voorstellen.

4.7.4 Meercomponentenbeelden

De methoden voor meercomponentenbeelden splitsen zich op in twee categorieën: die voor een gekende kleurenruimte en die voor een ongekende kleurenruimte. De technieken van de eerste categorie zijn enkel toepasbaar op RGB-beelden. De technieken van de tweede categorie zijn toepasbaar op alle kleurenruimtes, maar helaas zijn er geen implementaties publiek beschikbaar en zijn er in de literatuur maar zeer weinig resultaten gepubliceerd. We kunnen dan ook geen resultaten tonen voor de CMYK-beelden uit de testsets of voor

Tabel 4.10: Compressieresultaten van de technieken voor monochrome beelden op de componenten van "musicians-cmyk". Hieruit blijkt dat de K-component veel meer redundantie bevat.

techniek	C	M	Y	K	gem.
JBIG-S	4.91	4.98	5.29	4.03	4.80
LJPEG	4.87	4.89	5.18	4.38	4.83
FELICS	4.71	4.76	5.08	4.01	4.64
JPEG-LS	4.42	4.47	4.76	3.58	4.31
CALIC	4.26	4.31	4.58	3.44	4.15
BTPC	4.74	4.78	5.08	3.85	4.62
S+P-H	4.40	4.41	4.67	3.79	4.32
JPEG2000	4.41	4.44	4.69	3.65	4.30

multispectrale beelden.

Tabel 4.11 geeft een beperkt aantal resultaten weer van technieken zonder tonale decorrelatie, technieken met statische tonale decorrelatie en technieken met adaptieve tonale decorrelatie op een viertal testbeelden. De testbeelden kunnen bezwaarlijk representatief genoemd worden omdat enerzijds de spatiale resolutie van "lena-rgb" en "peppers-rgb" vrij laag is en omdat anderzijds de beelden "cats-rgb" en "water-rgb" een grote egaal zwarte achtergrond vertonen. De resultaten voor IB-CALIC, SICLIC en L-KLT zijn overgenomen uit de literatuur [12, 32, 257, 284]. De techniek "L-KLT" combineert de voorspelling van LJPEG met de verliesloze KLT op blokken van 50×50 [257].

Uit de tabel blijkt dat de tonale decorrelatie een wisselend effect heeft. De statische tonale decorrelatie is vrij effectief en kan oplopen tot 40% voor een beeld als "cats-rgb". Maar anderzijds kan dit ook een negatief resultaat hebben, zoals dat bijvoorbeeld het geval is voor "peppers-rgb". De winst van adaptieve tonale decorrelatie is niet wezenlijk groter dan van statische decorrelatie, maar het resultaat is wel consequent positief. Deze conclusie blijkt ook uit ander onderzoek in de literatuur [32, 284].

4.7.5 Snelheid

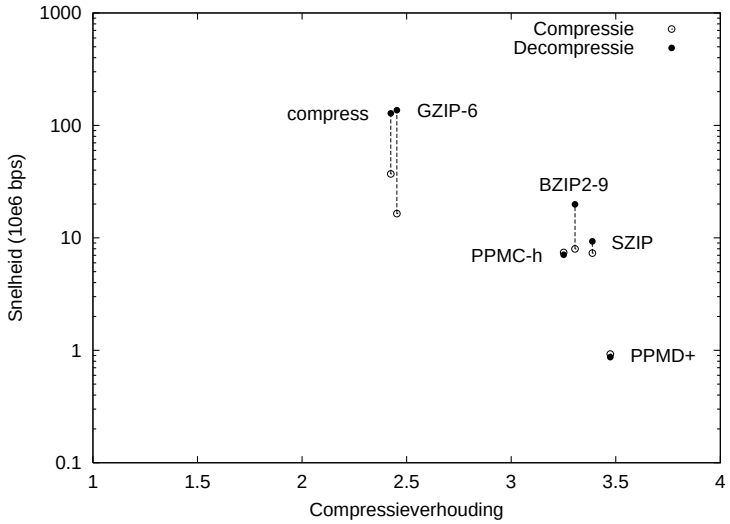
Het dynamisch bereik van de verwerkingssnelheid is veel groter dan dat van de compressieverhouding. Daar waar 10% winst in compressieverhouding al als significant beschouwd wordt, kan de snelheid over meerdere grootteordes variëren. Maar het vergelijken van de snelheid dient met de nodige omzichtigheid te gebeuren. Veel publiek beschikbare implementaties zijn helemaal niet

Tabel 4.11: Compressieresultaten van technieken voor meercomponentenbeelden voor een aantal RGB-beelden.

techniek	lena	peppers	cats	water	gem.
I - Zonder tonale decorrelatie					
LJPEG	4.90	5.22	3.69	2.62	4.11
JPEG-LS	4.52	4.74	2.61	1.81	3.42
CALIC	4.39	4.63	2.52	1.74	3.32
JPEG2000	4.59	4.88	2.56	1.80	3.46
II - Statische tonale decorrelatie					
JPEG-LS	4.59	4.92	1.89	1.47	3.22
JPEG2000	4.53	4.93	1.81	1.43	3.18
III - Adaptieve tonale decorrelatie					
IB-CALIC	–	–	1.81	1.51	–
SICLIC	4.46	–	1.86	1.45	–
L-KLT	4.51	5.04	1.96	1.52	–

geoptimaliseerd voor snelheid. En daarenboven is de snelheid afhankelijk van een groot aantal factoren, zoals processor, compiler, optimalisatie, cachegroottes, cachesnelheid, platform, grootte van het bestand en beschikbaar geheugen. Zo hebben een aantal PPM-gebaseerde technieken de negatieve eigenschap dat de snelheid afneemt naarmate het aantal verwerkte waarden en bijgevolg ook de grootte van het model toeneemt. De resultaten in deze paragraaf zijn indicatief voor wat haalbaar is met de verschillende aanpakken.

De individuele testbestanden zijn voldoende groot gekozen om het aandeel van het opstarten te minimaliseren en voldoende betrouwbaarheid te bekomen in de tijdsmetingen. De snelheid versus compressieverhouding wordt grafisch weergegeven voor “book1” (tekst van het Calgary corpus), “ro10k” (klassiek halftoonbeeld) en “musicians-c” (cyaancomponent). De snelheden zijn gemeten op een Intel-gebaseerd systeem (Pentium III, kloksnelheid 700 MHz, RAM geheugen 256 MB en cachegrootte 256 KB) met SUSE Linux 2.2.14 als besturingssysteem. Daar waar de broncode beschikbaar is, wordt als compiler gekozen voor Pentium GCC of PGCC (de GNU C-compiler geoptimaliseerd voor de Intel Pentium-processor) met de hoogste graad van optimalisatie, namelijk “-O6 -funroll-loops -mpentiumpro”. Voor compress, SZIP en JPEG2000 is de broncode niet beschikbaar en is de mate van optimalisatie specifiek voor het platform onbekend. Voor CALIC is enkel een binair programma voor SUN

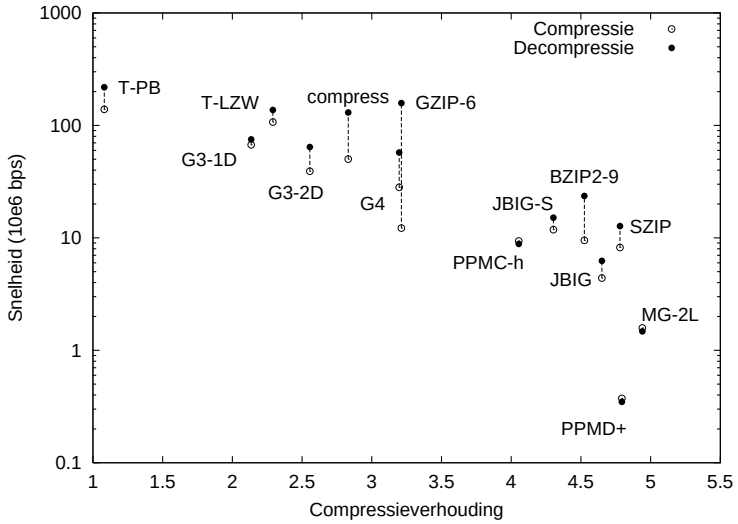


Figuur 4.20: Compressie- en decompressiesnelheid (in 10^6 bps) versus compressieverhouding voor het tekstbestand “book1”.

Solaris beschikbaar en de snelheden voor het standaardplatform worden teruggerekend naar het standaardplatform in evenredigheid met de resultaten voor JBIG. Indien meerdere implementaties beschikbaar zijn, geven de grafieken enkel de resultaten weer van de snelste implementatie. Voor compressietechnieken met instelbare parameters wordt geopteerd voor de standaardinstellingen. Nagenoeg altijd is de processor de snelheidsbeperkende component. De snelheid is consequent uitgedrukt in 10^6 bps (bits per seconde) volgens een logaritmische schaal. Het verschil tussen compressie- en decompressiesnelheid illustreert het al dan niet symmetrisch karakter van de techniek.

Figuur 4.20 geeft de compressie- en decompressiesnelheid versus de compressieverhouding weer van een aantal algemene technieken voor het tekstbestand “book1”. De LZ-gebaseerde technieken halen een decompressiesnelheid boven 100×10^6 bps en scoren daarmee het best qua snelheid. Technieken gebaseerd op bloksortering zoals BZIP2 en SZIP zijn veel efficiënter maar zijn grofweg één grootteorde trager. De bijkomende winst van PPM-gebaseerde technieken is vrij beperkt en brengt een aanzienlijke vertraging met zich mee.

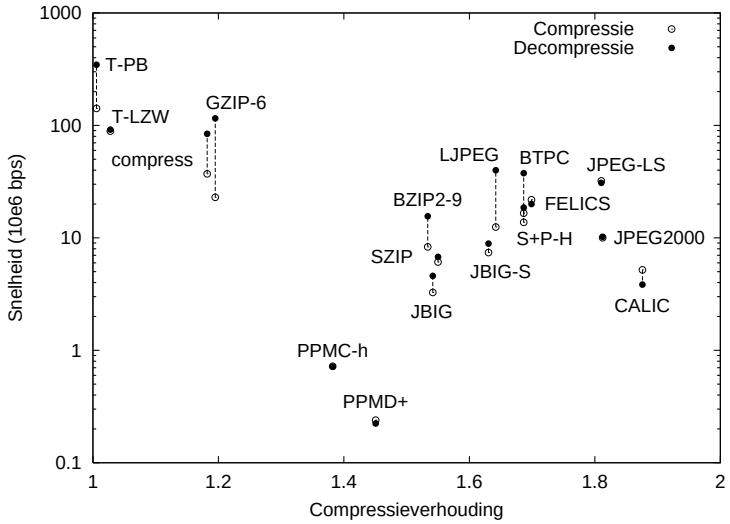
Voor binaire beelden is het moeilijker om een groot maar representatief testbeeld te kiezen. Omdat een groot deel van het verder onderzoek toegespitst is op gerasterde beelden, gaat de voorkeur hier naar het klassiek gerasterde testbeeld “mc-h-1270-ro” alias “ro10k”. Figuur 4.21 geeft de verwerkings-



Figuur 4.21: Compressie- en decompressiesnelheid (in 10^6 bps) versus compressieverhouding voor het klassiek gerasterde halftoonbeeld “ro10k”.

snelheid versus compressieverhouding weer van zowel algemene technieken als van techieken voor binaire beelden. Uit de figuur blijkt dat de algemene LZ-gebaseerde technieken heel hoge snelheden halen in combinatie met matige efficiëntie. De statische binaire technieken zoals G3 en G4 zijn heel snel maar hun efficiëntie is teleurstellend. De technieken gebaseerd op bloksortering scoren opvallend goed en combineren een behoorlijke snelheid met hoge efficiëntie. Hun snelheid is vergelijkbaar met JBIG-gebaseerde technieken en ongeveer een grootteorde trager dan de algemene technieken en de statische binaire technieken. De hoogste compressie wordt gehaald door de twee-niveau-contextgebaseerde techniek MG-2L, al is die te traag (circa 1.5×10^6 bps) voor de meeste praktische doeleinden. De snelheid van PPMD+ neemt af voor grotere bestanden en is dan ook onvoldoende in de praktijk, niettegenstaande de efficiëntie opmerkelijk hoog is.

De resultaten voor het monochrome beeld “musicians-c” worden weergegeven in figuur 4.22. Zoals verwacht halen de technieken voor monochrome beelden keer op keer de beste resultaten qua efficiëntie. Door het byte-georiënteerde karakter van de implementaties doen ze dit bovendien aan vrij hoge snelheden (grootteorde 30×10^6 bps). De techniek JPEG-LS combineert hoge snelheden met hoge efficiëntie en vormt hiermee een uitstekend compromis voor de meeste toepassingen. JPEG2000 haalt een vergelijkbare



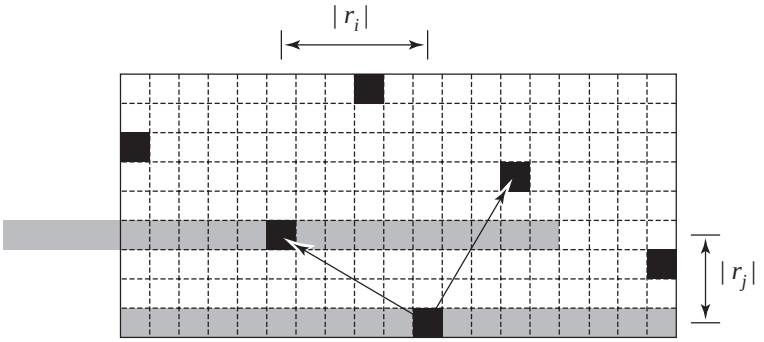
Figuur 4.22: Compressie- en decompressiesnelheid (in 10^6 bps) versus compressieverhouding voor de contone cyaancomponent "musicians-c".

efficiëntie, maar de toename aan flexibiliteit in de standaard zorgt ervoor dat de techniek nagenoeg driemaal trager is dan JPEG-LS. Een beperkte toename aan efficiëntie wordt bekomen door CALIC, maar dit gaat ten koste van ongeveer één grootteorde qua verwerkingssnelheid.

De voorgaande figuren illustreren slechts twee metrieken voor het evalueren van de bestaande technieken. Een derde eigenschap, de grootte van het vereiste werkgeheugen, is voor een aantal toepassingen van minstens even groot belang. Helaas is deze grootheid geen prioriteit bij het ontwikkelen van de meeste implementaties. Om deze reden zou een evaluatie van de technieken met betrekking tot deze eigenschap niet fair zijn. We gaan er dan ook niet verder op in.

4.8 Aanpassingen voor drukvoorbereidingsbeelden

Uit de resultaten blijkt dat een groot aantal verliesloze technieken niet optimaal scoren voor beelden uit de drukvoorbereidingsindustrie. We stellen een aantal uitbreidingen voor zodat deze technieken aanzienlijk betere resultaten behalen.



Figuur 4.23: Een aangepaste keuze van de referentielijn voor klassiek gerasterde halftoonbeelden komt de efficiëntie van Groep 4 ten goede. De onderste en de bovenste grijze lijnen stellen respectievelijk de codelijn en de aangepaste referentielijn voor.

4.8.1 Groep 4 met optimale referentielijn en huffmancodes

Overeenkomstig de standaard maakt Groep 4 gebruik van de voorgaande lijn als referentielijn. Dit is doorgaans de beste keuze voor tekstuele beelden omdat de correlatie tussen de lijnen monotoon zal afnemen naarmate de verticale afstand toeneemt. Bovendien zou ook een eventuele horizontale verschuiving van de referentielijn weinig zin hebben, omdat dit eveneens de correlatie negatief zou beïnvloeden.

De situatie voor klassiek gerasterde halftoonbeelden is helemaal anders. We stellen dan ook een aanpassing voor die een optimale referentielijn bepaalt voor elk beeld. De correlatie bereikt een maximum op een verplaatsing die overeenkomt met één van de basisvectoren (r_i, r_j) van het raster. Het heeft dan ook zin om als referentielijn de lijn te gebruiken die $|r_j|$ lijnen hoger ligt dan de codelijn [252]. Vervolgens wordt deze lijn horizontaal verschoven over een afstand $|r_i|$, zodat codelijn en referentielijn gealigneerd zijn. Figuur 4.23 illustreert deze semi-adaptieve tweedimensionale translatie van de referentielijn. De waarde van r_i en r_j worden als gekend verondersteld. De standaardinstellingen corresponderen met $(r_i, r_j) = (0, -1)$. De toename aan berekeningscomplexiteit beperkt zich tot het verschuiven van de referentielijn bij het begin van elke codelijn.

Een tweede aanpassing van deze standaard bestaat uit het gebruik van optimale huffmancodes in plaats van het statische codeboek [252]. Er komen in totaal drie huffmancodes aan te pas: één voor het coderen van de modus (horizontale, verticale of passeermodus), één voor de zwarte looplengtes in horizontale modus, en één voor de witte looplengtes in horizontale modus. Het

Tabel 4.12: De invloed van twee aanpassingen aan Groep 4 voor klassieke en stochastische halftoonbeelden. Compressieverhouding met gebruik van een optimale referentielijn ("-R") of optimale huffmancodes ("-H").

techniek	ro5k		ro10k		mo10k	
G4	3.50		3.20		1.38	
G4-R	4.43	(+27%)	3.85	(+20%)	–	(–)
G4-H	3.74	(+7%)	3.45	(+8%)	2.00	(+45%)
G4-RH	4.77	(+36%)	4.19	(+31%)	–	(–)

opstellen van een optimale huffmancode is een gekende techniek en kan heel snel verlopen. Het coderen van een beeld kan niet langer in één beweging gebeuren, wat een gepijplijnde implementatie in de weg staat. In tegenstelling tot het optimaliseren van de referentielijn, is deze aanpassing ook van toepassing op stochastische halftoonbeelden.

Tabel 4.12 toont een beperkt aantal resultaten op twee klassieke halftoonbeelden "ro5k" en "ro10k" en op een stochastisch halftoonbeeld "mo10k". Merk op dat de resultaten in de eerste lijn ("G4") overeenstemmen met de zwart-wit conventie die aanleiding geeft tot de hoogste efficiëntie. Voor de klassieke halftoonbeelden is het aanpassen van de referentielijn verantwoordelijk voor een aanzienlijke winst van ongeveer 25%. Het aanpassen van de huffmancodes geeft aanleiding tot een additionele winst van bijna 10%. Voor het stochastisch halftoonbeeld is de standaard referentielijn reeds optimaal. Het gebruik van optimale semi-adaptieve huffmancodes geeft evenwel aanleiding tot bijna 45% winst in efficiëntie. De verklaring hiervoor ligt in het feit dat de meeste pixels gecodeerd worden in horizontale modus met looplengtes die atypisch gedistribueerd zijn voor de gestandaardiseerde statische codes van Groep 4.

Al is de nettowinst van deze aanpassingen uitgesproken positief, het resultaat zal nooit in de buurt komen van technieken gebaseerd op contextmodellering zoals voorgesteld in tabel 4.8. Concrete snelheidsresultaten ontbreken, maar de impact van het aanpassen van de referentielijn op de verwerkingssnelheid is eerder beperkt. Het opstellen van optimale huffmancodes daarentegen vereist dat het beeld in twee bewegingen verwerkt wordt.

4.8.2 Statische tonale decorrelatie voor CMYK-beelden

Zoals eerder beschreven kunnen de methoden voor meercomponentenbeelden opgedeeld worden in twee categorieën naargelang de kleurenruimte gekend is

of ongekend. Technieken voor een gekende kleurenruimte veronderstellen allemaal de RGB-ruimte. De monochrome verliesloze compressie gebaseerd op spatiale voorspelling wordt voorafgegaan door een statische reversibele tonale transformatie. Het is ook mogelijk de tonale decorrelatie toe te passen na de spatiale voorspelling, maar voor de statistische codering. Daarom stellen we hier een techniek voor die we *Inter-color Error Prediction* of IEP noemen [54]. Deze aanpak is algemener omdat minder rekening gehouden wordt met de exacte interpretatie van de componenten. Bovendien is ze gemakkelijker uit te breiden voor beelden met meer dan drie componenten, zoals CMYK-beelden.

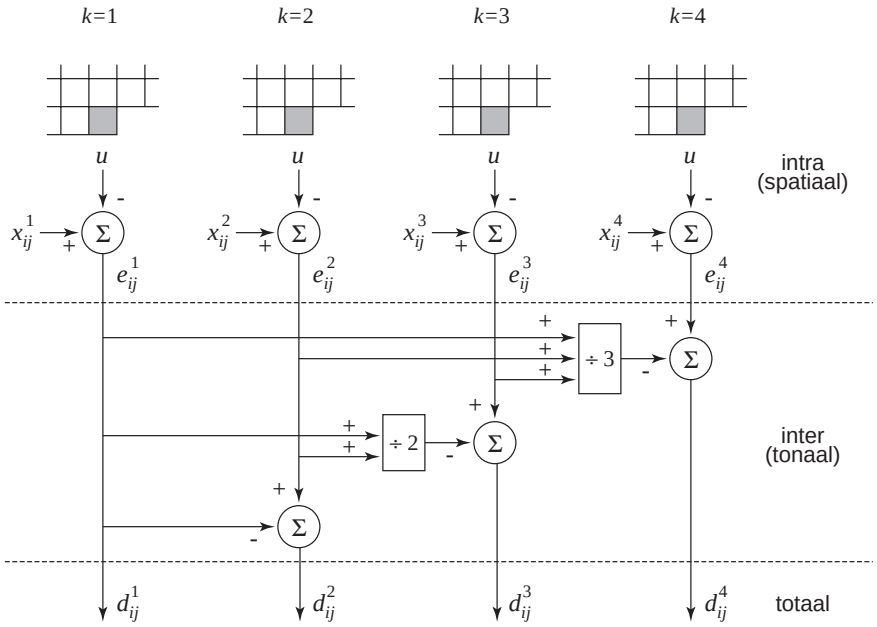
Eerst maakt de spatiale voorspeller u een schatting $\hat{x}_{ij}^k$ voor pixelwaarde x_{ij}^k gebaseerd op een causale omgeving binnen de component k. Hieruit volgt de voorlopige voorspellingsfout $e_{ij}^k = x_{ij}^k - \hat{x}_{ij}^k$ voor elke component k met $k = 1..n_c$, waarbij n_c het aantal componenten voorstelt. Na een eventuele herschikking van de volgorde van de componenten wordt de eerste component $k = 1$ als referentiecomponent genomen. Vervolgens corrigeert IEP de voorlopige voorspellingsfout e_{ij}^k van elke andere component $k = 2..n_c$ tot de definitieve voorspellingsfout

$$d_{ij}^k = e_{ij}^k - \left(\sum_{\alpha=1}^{k-1} e_{ij}^\alpha \right) \div (k-1),$$

met per definitie $d_{ij}^1 = e_{ij}^1$. De reversibiliteit van deze eenvoudige decorrelatie volgt rechtstreeks uit de constructie. Figuur 4.24 illustreert dit proces voor een beeld met $n_c = 4$ componenten.

De aanpassing is zowel toepasbaar op RGB-beelden als op CMYK-beelden, waar respectievelijk $n_c = 3$ en $n_c = 4$. De keuze van de referentiecomponent voor de RGB-kleurenruimte is arbitrair, al zal groen vaak het beste resultaat halen. Maar het ontstaansproces van CMYK-beelden heeft als gevolg dat de statistische afhankelijkheden tussen C, M, Y en K sterk kunnen variëren. Tabel 4.13 illustreert deze afhankelijkheid voor het beeld “musicians-cmyk”. Hieruit blijkt dat zwart (K) de beste keuze is voor de referentiecomponent. Dit is te verklaren door het feit dat zwart gemeenschappelijke informatie bevat van de oorspronkelijke kleuren uit het ontstaansproces. Bij andere beelden wordt hetzelfde gedrag vastgesteld.

De tonale decorrelatie van IEP is te combineren met om het even welke spatiale voorspeller. Tabel 4.14 geeft het resultaat weer van IEP op een aantal CMYK-beelden. De spatiale voorspeller is u_7 van LJPEG of de tweede-orde voorspeller van CALIC. Andere elementen van CALIC, zoals de binaire modus en de contextmodellering van de voorspellingsfout, zijn niet opgenomen. Uit de tabel volgt dat IEP consequent voor een winst zorgt van gemiddeld



Figuur 4.24: Inter-color Error Prediciton corrigeert de spatiale voorspellingsfout met de gemiddelde waarde van de voorgaande spatiale voorspellingsfouten.

14% voor de LJPEG-voorspeller en slechts 3% voor de CALIC-voorspeller. Dit resultaat wordt vergeleken met de decorrelatie door de Karhunen-Loève-transformatie (KLT) toe te passen op vierkante blokken (KLT-B) of op segmenten (KLT-S) [251, 257]. De techniek KLT-B doet het iets beter dan KLT-S. Het is opmerkelijk dat een eenvoudige aanpassing als IEP het bijna even goed doet als de rekenintensieve KLT-gebaseerde decorrelatie.

De aanpak van IEP heeft een aantal voordelen tegenover technieken die de tonale decorrelatie uitvoeren voor de spatiale decorrelatie, zoals JPEG-LS en JPEG2000 met de RCT. Zo is het aantal bewerkingen lager en moet het dynamisch bereik van de waarden niet uitgebreid worden. Maar er zijn ook nadelen aan verbonden. Omdat de tonale decorrelatie zijn plaats inneemt tussen de spatiale decorrelatie en de entropiecodering, dienen een aantal parameters van de standaardtechnieken aangepast te worden. Bovendien breekt deze operatie de interactie tussen voorspelling en contextmodellering van de voorspellingsfouten. Het is precies deze interactie die technieken als CALIC en JPEG-LS succesvol maakt. Deze breuk is niet onoverkomelijk, maar het zou een verregaande aanpassing vragen om CALIC of JPEG-LS te combineren met IEP.

Tabel 4.13: Voor "musicians-cmyk" haalt IEP consequent de beste compressie indien zwart (K) als referentiecomponent gekozen wordt. De kolom "h_0" stelt de empirische nulde-orde entropie voor per component (in bpp). Zonder gebruik van IEP bedraagt deze grootheid 4.73 bpp. De spatiale voorspeller is u_7 van LJPEG.

orde	h_0	orde	h_0	orde	h_0	orde	h_0
CMYK	4.34	MCYK	4.35	YCMK	4.44	KCMY	4.29
CMKY	4.38	MCKY	4.39	YCKM	4.43	KCYM	4.29
CYMK	4.36	MYCK	4.36	YMCK	4.44	KMCY	4.29
CYKM	4.35	MYKC	4.37	YMKC	4.45	KMYC	4.30
CKMY	4.40	MKCY	4.41	YKCM	4.46	KYCM	4.27
CKYM	4.40	MKYC	4.42	YKMC	4.47	KYMC	4.28
gem.	4.37	gem.	4.38	gem.	4.45	gem.	4.29

4.8.3 Visueel uniforme bijna-verliesloze beeldcompressie

Bijna-verliesloze compressietechnieken zijn verlieshebbende technieken die het geïntroduceerde verlies expliciet beperken. Meestal gebeurt dit door de voorspellingsfout van een verliesloze techniek te kwantiseren en aldus de maximale intensiteitsfout per pixel te beperken [4, 125, 266, 280]. Voor het gehele beeld stemt dit overeen met het beperken van het verlies aan de hand van een L_∞-criterium, dit in tegenstelling met DCT- en waveletgebaseerde coders die impliciet een L_2-criterium hanteren. Om de convergentie te bewaren dient de codeerstap dezelfde benaderde waarden $\tilde{x}$ gebruiken als de decodeerstap, een eigenschap die ook wel *synchronisatie* genoemd wordt. Het kwantiseren van de fout kan aanleiding geven tot een aantal artefacten, zoals *bandvorming* (*E.* banding), *granulaire ruis* (*E.* granular noise) en *randdrukte* (*E.* edge busyness) [182]. Een bijna-verliesloze variant van CALIC haalt uitstekende resultaten voor monochrome beelden [281].

In de CMYK-kleurenruimte krijgt deze uniforme kwantisatie een heel andere betekenis. De vier kleuren kunnen dan wel apart of gezamenlijk gekwantiseerd worden, maar geen van beide manieren stemt overeen met de subjectieve waarneming van kleur volgens het HVS. De CIEL*a*b*-ruimte werd ingevoerd om kleurverschillen af te meten volgens deze subjectieve metriek [106, 285]. De rechtstreekse niet-lineaire tranformatie van de CMYK-beelden naar de CIEL*a*b*-ruimte is evenwel niet aangewezen. Er treedt een groot verlies op aan tonale informatie door het irreversibele karakter van de 4D→3D transformatie en dit verlies is ontoelaatbaar in de niet-ideale wereld van de drukvoorbereidingsindustrie.

Tabel 4.14: Evaluatie van een aantal statische en adaptieve tonale decorrelatietechnieken voor CMYK-beelden. De spatiale voorspeller is gebaseerd op LJPEG of CALIC. De tonale voorspeller is statisch (IEP) of gebaseerd op de KLT, uitgevoerd op vierkante blokken (KLT-B) of op segmenten (KLT-S). De kolom "SCID*" stelt het gemiddelde voor over de 8 beelden van de ISO SCID testset.

tonaal	musicians	scid0	scid3	bike	cafe	woman	SCID*	gem.
I - Spatiale voorspeller u_7 van LJPEG								
(geen)	4.73	4.09	4.19	4.10	5.71	4.73	4.86	4.75
IEP	3.91	3.66	3.71	3.44	5.00	3.91	4.32	4.16
KLT-B	3.63	3.54	3.75	3.53	4.88	3.63	4.13	4.00
KLT-S	3.78	3.58	3.80	3.54	4.93	3.78	4.18	4.06
II - Spatiale tweede-orde voorspeller van CALIC								
(geen)	4.17	3.56	3.50	3.42	4.86	4.07	4.30	4.14
IEP	3.93	3.55	3.51	3.27	4.69	3.77	4.17	4.01
KLT-B	3.88	3.52	3.67	3.46	4.71	3.64	4.07	3.96
KLT-S	3.88	3.53	3.65	3.51	4.73	3.78	4.13	4.01

We stellen een oplossing voor die gebaseerd is op het lineariseren van de lokale CMYK→CIEL*a*b* transformatie aan de hand van een meerdimensionale taylorexpansie ter hoogte van de voorspelling [52, 56]. De resulterende jacobiaan wordt vervolgens ontbonden volgens de *singuliere-waardenontbinding* (*E.* singular value decomposition) of SVD [115, 180]. Hieruit volgt de lokale tonale decorrelatie en de schaling om een kwantisatie te bekomen die een uniform effect heeft in de CIEL*a*b*-ruimte. Opdat de jacobiaan een vierkante matrix zou zijn voegen we een fictieve vierde dimensie toe aan de CIEL*a*b*-ruimte. Het evalueren van de methode is niet evident omdat het verlies visueel niet te detecteren mag zijn. We bespreken elk van deze numerieke stappen in detail.

Conversie van CMYK naar CIEL*a*b*G De conversie van een kleur (c, m, y, k) in de CMYK-ruimte naar CIEL*a*b* verloopt via de XYZ-waarden van de kleur en van het gekozen witpunt [285]. Er zijn meerdere manieren van de CMYK-ruimte over te gaan naar de XYZ-ruimte. Omdat de fysische eigenschappen van de inkt, het papier, het rasterproces en de belichting als onbekend beschouwd worden, kiezen we voor een vereenvoudigde aanpak. Stelt (c, m, y, k) de kleur voor waarbij elke waarde in het interval

[0, 255] ligt, dan zijn de tristimuluswaarden (r, g, b) gegeven door

$$\begin{aligned} r &= \max(255 - c - k, 0), \\ g &= \max(255 - m - k, 0), \\ b &= \max(255 - y - k, 0). \end{aligned}$$

Keuze van een D_{65} lichtbron resulteert in de XYZ-waarden

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \cdot \begin{bmatrix} r \\ g \\ b \end{bmatrix},$$

waaruit vervolgens de CIEL*a*b*-waarden (l, a, b) berekend kunnen worden volgens standaardformules. Deze transformatie van (c, m, y, k) naar (l, a, b) is irreversibel omdat ze een 4D-ruimte afbeeldt op een 3D-ruimte en omdat grote oppervlakken van de CMYK-ruimte afgebeeld worden op identieke (r, g, b)-tripletten.

De transformatie verwijdert de relevante informatie van de *grijswaardenvervanging* (*E.* gray color replacement) of GCR. Om deze reden voegen we een artificiële vierde dimensie G toe aan de CIEL*a*b*-ruimte, volgens

$$g = \alpha(c + m + y - 3k),$$

waarbij $\alpha = 100/(3 \times 255)$ als normalisatieconstante optreedt. Het invoeren van deze nieuwe grootheid resulteert in een vierdimensionale CIEL*a*b*G-ruimte, waarbij de waarde van g beperkt is tot het interval $[-100, 100]$.

Linearisatie in de omgeving van de voorspelling Stel dat $\mathbf{x} = (c, m, y, k)$ de kleurvector voorstelt in de CMYK-ruimte, en dat de vectorfunctie $\bar{F}$ de transformatie beschrijft naar de CIEL*a*b*G-ruimte. Bij reconstructie is niet de oorspronkelijke waarde $\mathbf{x}$ maar enkel de benaderde waarde $\tilde{\mathbf{x}}$ gekend. Synchronisatie vereist dat de voorspelling $\hat{\mathbf{x}}$ enkel gebruik maakt van benaderde waarden $\tilde{\mathbf{y}}$.

Het visueel kleurverschil tussen de oospronkelijke pixelwaarde $\mathbf{x}$ en de gereconstrueerde pixelwaarde $\tilde{\mathbf{x}}$ wordt gegeven door $\Delta E = \|\bar{F}(\mathbf{x}) - \bar{F}(\tilde{\mathbf{x}})\|$, waarbij $\|\cdot\|$ de L_2-norm voorstelt. Een eerste-orde afbreking van de taylorreeksontwikkeling rond de voorspelling $\hat{\mathbf{x}}$ geeft

$$\begin{aligned} \|\bar{F}(\mathbf{x}) - \bar{F}(\tilde{\mathbf{x}})\| &= \|\bar{F}(\hat{\mathbf{x}} + \mathbf{d}) - \bar{F}(\hat{\mathbf{x}} + \tilde{\mathbf{d}})\| \\ &\approx \left\| \frac{\partial \mathbf{F}}{\partial \mathbf{x}} \cdot (\mathbf{d} - \tilde{\mathbf{d}}) \right\| \\ &\approx \left\| \mathbf{J} \cdot (\mathbf{d} - \tilde{\mathbf{d}}) \right\| \end{aligned}$$

onder de randvoorwaarde dat de voorspellingsfouten $\mathbf{d}$ en $\tilde{\mathbf{d}}$ voldoende klein zijn. Hierin stelt $\mathbf{J}$ de *jacobiaan* voor van de kleurtransformatie. Toegepast op de nieuwe CIEL*a*b*G-ruimte levert dit

$$\mathbf{J} = \begin{bmatrix} \partial l/\partial c & \partial l/\partial m & \partial l/\partial y & \partial l/\partial k \\ \partial a/\partial c & \partial a/\partial m & \partial a/\partial y & \partial a/\partial k \\ \partial b/\partial c & \partial b/\partial m & \partial b/\partial y & \partial b/\partial k \\ \alpha & \alpha & \alpha & -3\alpha \end{bmatrix},$$

met de partiële afgeleiden geëvalueerd in $\hat{\mathbf{x}}$. Gelet op de irreversibiliteit aan de randen van de CMYK-kubus, zal het dikwijls voorkomen dat deze partiële afgeleiden nul opleveren. Dit resulteert in een singuliere matrix en er zullen maatregelen nodig zijn om dit te vermijden.

SVD van de jacobiaan De vierkante matrix $\mathbf{J}$ kan ontbonden worden als

$$\mathbf{J} = \mathbf{U} \cdot \mathbf{D} \cdot \mathbf{V}^{\mathsf{T}},$$

met $\mathbf{D}$ een diagonaalmatrix die de singuliere waarden voorstelt en $\mathbf{U}$ en $\mathbf{V}$ orthogonale matrices (dus $\mathbf{U}^{\mathsf{T}} = \mathbf{U}^{-1}$ and $\mathbf{V}^{\mathsf{T}} = \mathbf{V}^{-1}$). Deze ontbinding heet de SVD en kan altijd berekend worden. Als de matrix $\mathbf{J}$ singulier is, zullen een aantal diagonaalelementen van $\mathbf{D}$ nul zijn.

De bewerking $\mathbf{J} \cdot \mathbf{u}$ stemt voor een vector $\mathbf{u}$ overeen met achtereenvolgens: (1) een passieve rotatie volgens $\mathbf{V}^{\mathsf{T}}$ tot de assen van maximale visuele gevoeligheid; (2) een anisotrope dilatatie volgens de diagonaalelementen van $\mathbf{D}$ overeenkomstig deze gevoeligheden en (3) een passieve rotatie volgens $\mathbf{U}$ tot de assen van de CIEL*a*b*G-ruimte. Merk op dat een rotatie *passief* (*E.* passive) genoemd wordt indien het coördinatensysteem roteert maar de eigenlijke vectoren ongewijzigd blijven. Deze laatste transformatie is orthogonaal en kan verwaarloosd worden voor het berekenen van de norm in

$$\begin{aligned} \|\bar{F}(\mathbf{x}) - \bar{F}(\tilde{\mathbf{x}})\| &\approx \|\mathbf{U} \cdot \mathbf{D} \cdot \mathbf{V}^{\mathsf{T}} \cdot (\mathbf{d} - \tilde{\mathbf{d}})\| \\ &\approx \|\mathbf{D} \cdot (\mathbf{V}^{\mathsf{T}} \cdot \mathbf{d} - \mathbf{V}^{\mathsf{T}} \cdot \tilde{\mathbf{d}})\|. \end{aligned}$$

Deze laatste vergelijking ligt aan de basis van de voorgestelde kwantisatieprocedure: (1) roteer de voorspellingsfout $\mathbf{d} = \mathbf{x} - \hat{\mathbf{x}}$ volgens $\mathbf{V}^{\mathsf{T}}$ tot de visuele hoofdassen, zodat $\mathbf{d}' = \mathbf{V}^{\mathsf{T}} \cdot \mathbf{d}$; (2) schaal de componenten van $\mathbf{d}'$ overeenkomstig de singuliere waarden (een hoge singuliere waarde betekent een hoge gevoeligheid), zodat $\mathbf{d}'' = \mathbf{D} \cdot \mathbf{d}'$; (3) kwantiseer $\mathbf{d}''$ uniform in de vier dimensies (en codeer de gekwantiseerde waarden aan de hand van een entropiecoder), zodat $\tilde{\mathbf{d}}'' = Q(\mathbf{d}'')$ waarbij $Q(\cdot)$ de uniforme kwantisatie voorstelt; (4) schaal het gekwantiseerde resultaat in de omgekeerde zin, zodat $\tilde{\mathbf{d}}' = \mathbf{D}^{-1} \cdot \tilde{\mathbf{d}}''$; (5)

roteer in omgekeerde zin, zodat $\tilde{\mathbf{d}} = \mathbf{V} \cdot \tilde{\mathbf{d}}'$; en tenslotte (6), synchroniseer de benaderde waarde, zodat $\tilde{\mathbf{x}} = \hat{\mathbf{x}} + \tilde{\mathbf{d}}$. Omdat $\mathbf{D}$ singulier kan zijn, wordt elk diagonaalelement d vervangen door $\max(d, d_0)$, met d_0 een kleine maar constante waarde (0.1 in onze implementatie).

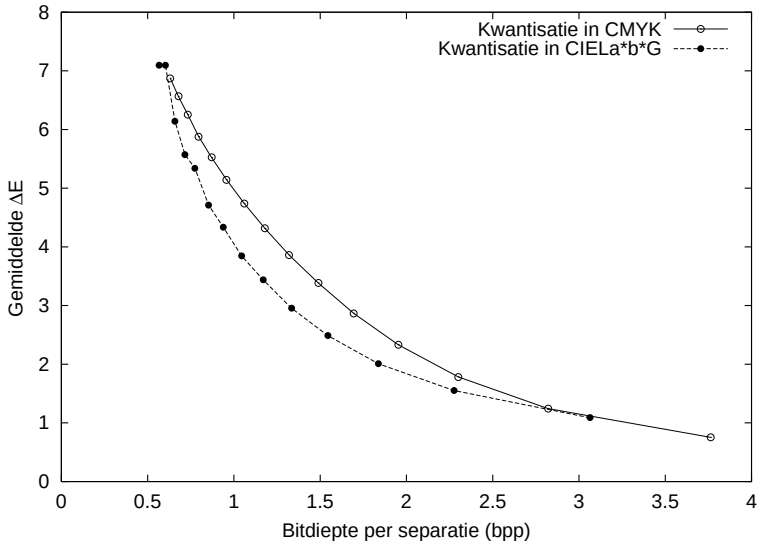
Door het optreden van singulariteiten in de transformatie gaat de gebruikte taylorexpansie niet altijd op. Bijgevolg kan deze methode strikt genomen niet als een bijna-verliesloze methode beschouwd worden. Daarenboven is de eerste-orde-benadering slechts zinvol indien de voorspellingsfouten klein zijn, wat zeker niet altijd het geval is. Maar omdat het maskeringseffect terzelfdertijd zal optreden als wanneer er zich grote voorspellingsfouten voordoen, neemt de impact van deze afwijking af. Het *maskeringseffect* (*E.* masking effect) van randen betreft de eigenschap van het HVS dat kleine beeldafwijkingen onopgemerkt blijven in de buurt van grote beeldvariaties.

Resultaten Figuur 4.25 geeft de gemiddelde CIEL*a*b* ΔE versus de bitdiepte weer voor het standaard CMYK testbeeld “musicians-cmyk”. Hieruit volgt duidelijk dat de visueel gewogen kleurfout aanzienlijk afneemt als gevolg van de verfijnde kwantisatieprocedure. Het effect is het grootst in het gebied van 0.5 tot 2.5 bpp, wat overeenstemt met een compressieverhouding van ongeveer 3 tot 15. Precies in dit gebied bewijzen de bijna-verliesloze technieken hun nut. De winst voor een aantal andere beelden is iets kleiner maar eveneens positief.

Het uitvoeren van deze kwantisatieprocedure voor elke pixel vraagt om enorm veel rekenwerk. Het zou mogelijk zijn de CMYK-ruimte te partitioneren en de lokale rotaties en visuele gevoeligheden te tabelleren. De waarde van deze aanpak blijft evenwel beperkt door de gebrekkige modellering van het HVS. Een geavanceerde modellering moet niet alleen rekening houden met de tonale maar ook met de spatiale en frequentiegevoelige eigenschappen van de menselijke visuele waarneming.

4.9 Samenvatting en eigen bijdragen

Het domein van de verliesloze beeldcompressie is geworteld in de verliesloze compressie van teksten en andere eendimensionale sequenties. Een correcte interpretatie van de beeldinformatie, zoals het tweedimensionaal karakter, de kwantitatieve aard van de pixelintensiteit en de woordbreedte van de pixelwaarde, leidt automatisch tot een aantal aanpassingen. De recent gepubliceerde standaarden tonen aan hoe de focus verschuift van steeds betere en/of snellere compressie naar meer flexibiliteit. Bovendien specificeert een standaard enkel het gecomprimeerd formaat en dit genereert een grote mate aan



Figuur 4.25: Invloed van de visueel uniforme bijna-verliesloze kwantisatie voor het beeld "musicians-cmyk". De gemiddelde CIEL*a*b* ΔE fout is uitgedrukt tegenover de bitdiepte per component na compressie (in bpp).

ontwerp- en implementatievrijheid.

Algemene technieken kunnen grofweg opgedeeld worden in twee categorieën: karaktergebaseerde methoden en blokgebaseerde methoden. De karaktergebaseerde methoden bouwen expliciet een statistisch model op dat voor elk pixel een conditionele waarschijnlijkheidsschatting genereert. Deze schattingen zijn gebaseerd op contexten in het verleden en dienen als invoer voor een entropiecoder. Blokgebaseerde technieken segmenteren de toekomstige evolutie in stukken van min of meer gelijke waarschijnlijkheid. Het statistisch model is impliciet aanwezig en de verwerkingssnelheid ligt veel hoger. De modelvorming van algemene technieken kan vrij complex zijn. De contextmodellen combineren een hoge efficiëntie met een hoge mate aan flexibiliteit.

De historisch oudste contextmodellen waren ontworpen voor binaire beelden omdat deze slechts één bit per bronwaarde vereisen. De meest geavanceerde standaarden, JBIG en JBIG2, zijn dan ook gebaseerd op contextmodellering. Om de flexibiliteit te verhogen, worden daar multiresolutiedecompositie en patroonherkenning aan toegevoegd. Bij het overgaan van tekstcompressie naar beeldcompressie is het leerrijk om het veralgemeend compressieschema uit te breiden en te veralgemenen met een aantal specifieke entiteiten.

Methoden voor monochrome beelden zijn gebaseerd op sequentiële voor-

spelling, op multiresolutiedecompositie of ze zijn blokgebaseerd. De meest efficiënte technieken, zoals JPEG-LS en CALIC, maken gebruik van niet-lineaire voorspelling en contextmodellen zowel voor de voorspelling als voor de foutmodellering. De nieuwste standaard, JPEG2000, focust vooral op flexibiliteit en is om die reden gebaseerd op multiresolutiedecompositie aan de hand van wavelets. Technieken voor meercomponentenbeelden en meerdimensionale beelden zijn een logische voortzetting van dezelfde basisprincipes.

De experimentele resultaten tonen hoe verliesloze compressie tekstbestanden tot 4 maal kleiner kunnen voorstellen. Voor binaire beelden kan dit veel meer zijn en is het dynamisch bereik ook veel groter. Halftoonbeelden hebben atypische statistieken en hierop falen de meeste statische methoden. Uit de resultaten blijkt dat contextmodellering de aangewezen richting is, maar er kan zeker vooruitgang geboekt worden. Voor monochrome beelden is een compressiefactor 2 al een uitstekend resultaat. Het uitbuiten van kleur geeft een bijkomende winst van ongeveer 10% tot 20%. Snelheidsmetingen tonen aan hoe een beperkte toename in compressieverhouding al vlug één of meerdere grootteordes in snelheid kan kosten.

Omdat halftoonbeelden en CMYK-beelden niet tot de doelcategorie horen bij de meeste technieken, stellen we een aantal aanpassingen voor. Zo levert het optimaliseren van de referentielijn en de huffmancodes bij Groep 4 een winst op van ongeveer 35% bij halftoonbeelden. Een eenvoudig uitbreidbare statische tonale decorrelatie bij CMYK-beelden levert een winst op van ongeveer 10%, wat in de buurt komt van de computationeel veel intensievere aanpak gebaseerd op de KLT. Tenslotte stellen we een techniek voor die vergelijkbaar is met bijna-verliesloze compressie, maar die als criterium het subjectief gewogen kleurverschil in de CIEL*a*b*-ruimte hanteert.

Het nieuwe paradigma voor de analyse van verliesloze beeldcompressietechnieken is eveneens een eigen bijdrage. Het laat toe een techniek op te splitsen in de opbouwende componenten. Het onderzoek naar de state of the art voor beelden in de drukvoorbereidingsindustrie en de medische wereld heeft geresulteerd in een aantal bijdragen op internationale conferenties en één gedeelde publicatie in een internationaal tijdschrift [49, 60, 61, 65, 66, 176, 178, 254]. De aanpassingen van de bestaande technieken voor beelden uit de drukvoorbereidingsindustrie zijn eveneens voorgesteld op internationale conferenties [52, 54, 56, 251, 252].

Hoofdstuk 5

Contextmodellering van halftoonbeelden

5.1 Inleiding

De state of the art van verliesloze beeldcompressie toont aan hoe JBIG hoge compressiefactoren haalt voor binaire tekstbeelden door gebruik te maken van contextmodellering. Deze modelvorming is zeer efficiënt, heel flexibel en bovendien verloopt ze aan een behoorlijke snelheid door gebruik te maken van de QM-coder als entropiecoder.

Maar voor halftoonbeelden zijn de resultaten niet altijd eenduidig. Een vergelijking van de resultaten van JBIG met die van enkele geavanceerde algemene technieken zoals BZIP2 en PPMD+ laat vermoeden dat een aanpassing aan de specifieke eigenschappen van halftoonbeelden de efficiëntie aanzienlijk kan verhogen. In dit hoofdstuk onderzoeken we het effect van een adaptieve sjabloonkeuze en de impact hiervan op de snelheid van een geoptimaliseerde software- en hardware-implementatie. Het valt op te merken dat de halftoonbeelden in een opgemaakte pagina de moeilijkst te comprimeren elementen zijn en dan ook de nodige aandacht verdienen.

De vraag kan terecht gesteld worden naar de bestaansreden van verliesloze compressie van halftoonbeelden. In veel gevallen is het inderdaad aangewezen om het oorspronkelijke contone beeld te comprimeren. Is het halftoonbeeld nodig, dan volstaat het het contone beeld te decomprimeren en opnieuw te rasteren. Op deze manier blijft de maximale kwaliteit bewaard en kan het rasterproces geoptimaliseerd worden aan het uitvoermedium naar keuze.

Toch zijn er een aantal evoluties waar te nemen in de drukvoorbereidingsindustrie waar er nood is aan tijdelijke opslag en snelle verzending van halftoonbeelden [232]. In het geval van *verdeel-en-print* (*E.* distribute-and-print)

stuurt een centrale eenheid een document naar een aantal drukmachines die geografisch verdeeld zijn. Distributie van de halftoonbeelden vermijdt dat het vaak rekenintensieve halftoonproces meermalig moet uitgevoerd worden. Bij *drukken op aanvraag* (*E.* print on demand) of *boek-op-aanvraag* (*E.* book on demand) worden de documenten opgeslagen in een digitaal formaat en het drukken gebeurt op de plaats en het tijdstip van aanvraag. Maar ook lokale operaties kunnen gebruik maken van halftooncompressie. Bij *variabele-data-drukwerk* (*E.* variable data printing) of *gepersonaliseerd drukwerk* (*E.* personalized printing) is er doorgaans maar een klein verschil tussen de pagina's en het is dan ook nutteloos om de hele pagina telkens opnieuw volledig te rasteren. Hogeresolutieprinters zoals 1200×1200 dpi vierkleurenprinters kunnen halftooncompressie gebruiken om de vereiste buffercapaciteit te verkleinen. Tenslotte vormen hogeresolutiefaxen een ander potentieel toepassingsgebied waar vooral JBIG zich op concentreert.

Tegenover de state of the art van het vorige hoofdstuk lijken een aantal bijdragen in dit hoofdstuk soms voor de hand liggend. Maar zoals reeds vermeld in de inleiding van het vorige hoofdstuk moet er rekening gehouden worden met de wisselende tijdslijn. De kern van het onderzoek voor deze bijdragen vond plaats in de periode van 1996 tot 2000. Dit is enkele jaren voor de publicatie van JBIG2, de meest recent verschenen compressiestandaard voor binaire beelden.

Paragraaf 5.2 gaat dieper in op het halftoonproces dat verantwoordelijk is voor de atypische statistieken van halftoonbeelden. Hierop voortbouwend beschrijft paragraaf 5.3 hoe een semi-adaptieve keuze van het sjabloon voor de hand ligt. Paragraaf 5.4 toont aan hoe de gevolgen voor de verwerkingssnelheid geminimaliseerd kunnen worden door gebruik te maken van een aangepaste implementatie. Dat de efficiëntie nog verder kan verhoogd worden door gebruik te maken van een variabele-orde-model wordt aangetoond in paragraaf 5.5. Vervolgens breidt paragraaf 5.6 deze techniek uit naar residucodering voor monochrome en meercomponentenbeelden en toont hiermee de universaliteit aan van de voorgestelde aanpak. Tenslotte vat paragraaf 5.7 de belangrijkste resultaten en de eigen bijdragen samen.

5.2 Overzicht van halftoonproces

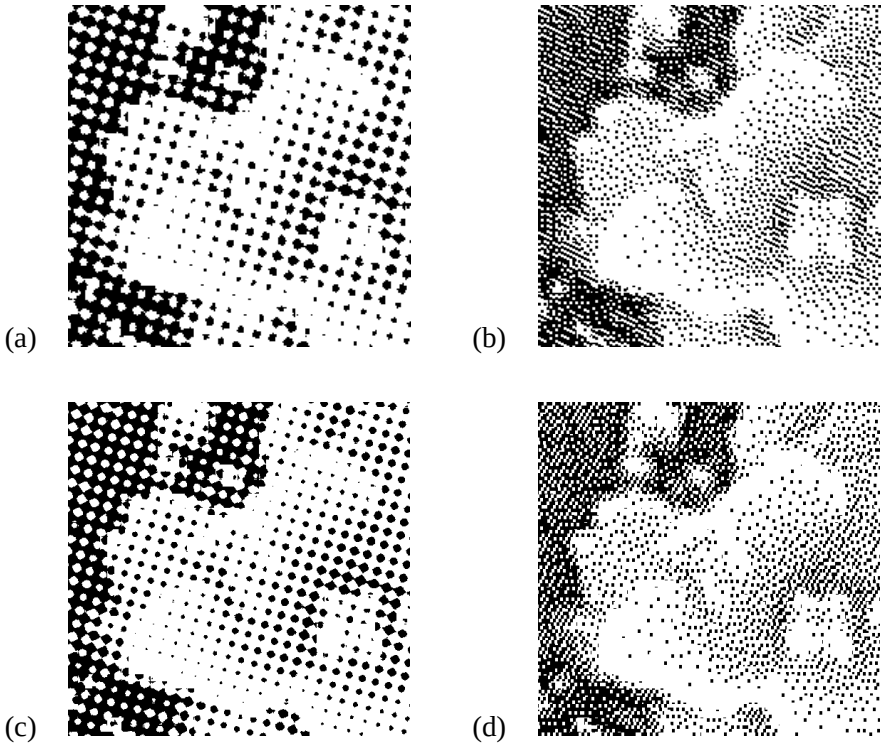
Het doel van het *halftoonproces* (*E.* halftoning) of *rasterproces* (*E.* screening) is om een monochroom beeld te vervangen door een patroon van *stipjes* of *dots* (*E.* dots) op een dusdanige manier dat de visuele indruk ongewijzigd blijft [247]. Een halftoonbeeld bestaat uit een groot aantal van deze dots die de latere inktvlekjes voorstellen. De dots kunnen rond, vierkant, cirkelvormig,

elliptisch of ruitvormig zijn en een belichter licht elk van deze dots uit op film aan de hand van microscopische *laserpunten* (*E.* spots). De fysische resolutie van het uitvoermedium wordt uitgedrukt in dots per inch (dpi).

Halftoontechnieken zijn onder te brengen in twee gescheiden categorieën. Bij *klassiek rasteren* (*E.* classical halftoning) variëren de inviduele dots qua grootte en zijn ze gepositioneerd volgens een regelmatig rechthoekig rooster geroteerd over een bepaalde *rasterhoek* (*E.* screening angle). De spatiale periodiciteit ligt vast en deze *halftoonresolutie* (*E.* halftone resolution) of *lijnfrequentie* (*E.* ruling) wordt uitgedrukt in lijnen per inch (lpi). De beeldinformatie wordt gecodeerd door de amplitude van de rasterdots te moduleren, vandaar ook de naam *amplitudemodulatie* of AM. Als daarentegen de grootte van de individuele dots constant is maar de positie ervan varieert, wordt de term *stochastisch rasteren* (*E.* stochastic screening) gehanteerd. De beeldinformatie wordt nu gecodeerd door de lokale frequentie te moduleren, vandaar de term *frequentiemodulatie* of FM. Figuur 5.1 illustreert het resultaat van dit rasterproces aan de hand van uitvergrotingen van hetzelfde detail. Een nog sterkere uitvergroting waarop de individuele dotstructuren duidelijk zichtbaar zijn is te zien in figuur A.5 in appendix A.

Het is essentieel dat het halftoonproces de oorspronkelijke grijswaarde zo getrouw mogelijk reproduceert, zonder dat er artefacten toevoegd worden. Voornaamste bron van artefacten is de interactie van de niet te onderscheiden kleine dots tot visueel storende macrostructuren. Voor klassiek rasteren komt dit neer op *frequentieverwarring* (*E.* aliasing) of *moiré*, dit zijn laagfrequente periodische structuren als gevolg van superpositie van meerdere hoogfrequente roosters. Voor stochastisch rasteren is dit vooral het optreden van *wormen* (*E.* worms), dit zijn gelijklopende geconnecteerde gebieden.

Bij vierkleurendruk zal het klassiek halftoonproces de vier roosters roteren over verschillende hoeken om het risico op moiré te beperken. Dit criterium is eenvoudig uit te drukken in het frequentiedomein: lineaire combinaties van de basisvectoren van de reciproque roosters moeten uit de buurt van de oorsprong blijven. Concreet blijkt dat het moiré-effect als minder storend of zelfs aangenaam wordt ervaren als de onderlinge hoek tussen de verschillende roosters 30° bedraagt. De resulterende patronen worden *rosetten* genoemd. Bovendien is de frequentiegevoeligheid van het menselijk visueel systeem het laagst rond een hoek van 45°. Dit resulteert in de volgende hoeken: zwart op 45°, cyaan en magenta op 75° en 15° en tenslotte geel op 0°. Merk op dat de hoek tussen geel en cyaan of magenta slecht 15° is, maar dit is geen probleem wegens de specifieke eigenschappen van gele inkt. Het energetisch vermogen van gele inkt om licht te absorberen is immers aanzienlijk lager dan dat van magenta of cyaan wat resulteert in een lage dekkingsgraad. Daarenboven is



Figuur 5.1: Uitvergroting van hetzelfde detail van het "musicians-c" beeld: (a) klassiek rasteren aan 110 lpi en 1270 dpi, (b) stochastisch rasteren aan 40μ en 1270 dpi, (c) klassiek rasteren aan 135 lpi en 2540 dpi en (d) stochastisch rasteren aan 40μ en 2540 dpi.

de spatiale resolutie van het HVS voor gele inkt heel laag. De tangens van 15° en 75° is irrationaal en dit heeft een aantal gevolgen voor de halftoonbeelden [70, 123]. Ofwel zijn de componenten van de basisvector van het rooster niet geheel. Ofwel zal gekozen worden voor lichtjes afwijkende hoeken en lichtjes afwijkende lijnfrequenties. In ieder geval is het halftoonbeeld niet zomaar als een mathematisch geheel periodisch rooster te interpreteren. Dit kan eventuele compressietechnieken gebaseerd op ontrasteren aanzienlijk bemoeilijken. Waarden voor de resolutie r_p van het uitvoermedium variëren van 600 tot 5000 dpi. De halftoonresolutie r_h varieert van 40 tot 300 lpi. Theoretisch gezien wordt het aantal verschillende grijstinten gegeven door $(r_p/r_h)^2 + 1$. De beeldresolutie moet minstens 1.5 tot 2 maal zo groot zijn als de halftoonresolutie.

Bij stochastisch rasteren wordt de grootte van de halftoondots beperkt door

de fysische eigenschappen van het inkttype, het uitvoermedium en de drager. De dots worden zo klein mogelijk gekozen onder de voorwaarde dat het drukproces ze betrouwbaar kan reproduceren. Figuur 5.1(b) en 5.1(d) illustreren hoe een verhoging van de resolutie niet resulteert in steeds kleiner wordende dots, maar wel in een fijnere positionering ervan. Onder minimale omstandigheden stemt één enkele dot overeen met 1×1 laserspots. Voor drukwerk aan hogere kwaliteit neemt de dotgrootte toe tot 2×2 of zelfs 4×4 laserspots, waarbij de resolutie van het doelrooster telkens verdubbelt. Deze basisdots vormen een belangrijke eigenschap die door een verliesloos compressiealgoritme uitgebuit kan worden.

Rasteren aan hoge resolutie geeft aanleiding tot een aanzienlijke expansie van de data. Zo heeft een 2540 dpi halftoonbeeld bijna 9 maal meer opslagruimte nodig dan het originele 300 dpi contone beeld. Efficiënte compressie gekoppeld aan voldoend hoge verwerkingssnelheden kan hier een oplossing bieden. De afmetingen van de doorsnee halftoonbeelden zijn veel groter dan van de klassieke binaire tekstbeelden. De aandacht gaat hierna vooral naar klassieke halftoonbeelden omdat deze het meest voorkomen.

5.3 Keuze van het contextsjabloon

Statistisch is de waarde van een halftoonpixel sterk gerelateerd aan de relatieve ligging (de spatiale fase) van die pixel ten opzichte van het doelrooster. Zo zal een centrale pixel nagenoeg altijd zwart zijn en is de pixel op een halve periode nagenoeg altijd wit. Een contextmodel dat de gediscretiseerde fase opneemt in de conditionerende contexten lijkt dan ook een goede keuze. Maar omdat de basisvector van het doelrooster niet altijd gehele componenten heeft, en omdat de rasterhoek en halftoonfrequentie licht kunnen afwijken van de vooropgestelde waarden, kan het gebruik van een dergelijke globale conditionerende grootheid tegenvallen. Deze globale gevoeligheid is het zwakke punt van elke compressietechniek die zich op ontrasteren baseert.

Het is dan ook aangewezen om de contexten enkel uit lokale conditionerende grootheden op te bouwen. Intuïtief betekent “lokaal” niet verder dan enkele halftoondots. Zo vormt de waarde van de pixel die op exact één basisvector gelegen is een uitstekende kandidaat om als conditionerende grootheid te fungeren. Nu kan als vuistregel gesteld worden dat de zijde van één halftoonbasiscel ongeveer 16 pixels moet bedragen om tot 256 verschillende grijstinten te komen. Ter vergelijking bedraagt de maximale L_1-afstand van de vaste sjabloonpixels in de bodemlaag van JBIG respectievelijk 3 en 4 pixels voor het sjabloon dat 3 en 2 lijnen telt (merk op dat de AT-pixel niet meegerekend wordt). Het $22/10$-sjabloon van het MG twee-niveau-contextmodel reikt

verder, maar de maximale afstand blijft beperkt tot 5 pixels. Ter referentie zijn deze sjablonen weergegeven in figuur 4.9 en 4.8. Deze sterke lokaliteit van de bodemlaag is er verantwoordelijk voor dat JBIG betere resultaten haalt in multiresolutiemodus dan in sequentiële modus, zelfs indien een AT-pixel toegelaten wordt die zich tot 8 pixels ver mag bevinden.

Naast de positie van de sjabloonpixels is ook de grootte van het contextsjabloon een belangrijke parameter. Omdat het aantal pixels veel groter is, is het risico op contextverdunning veel lager en zal de optimale orde hoger liggen.

5.3.1 Optimalisatie van het JBIG-sjabloon

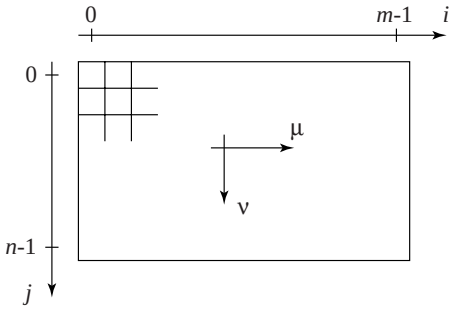
De JBIG-standaard definieert één AT-pixel in het sjabloon van de bodemlaag [118]. De relatieve positie van die pixel is volledig adaptief en kan wijzigen aan het begin van elke lijn. De standaard laat strikt genomen een relatieve positie (μ, ν) toe tot 127 pixels in horizontale richting en 255 pixels in verticale richting. Maar in de beschrijving van de minimale ondersteuning van de *basisimplementatie* (*E.* baseline implementation) beperkt de standaard tegelijkertijd deze waarden tot 8 en 16 pixels in horizontale richting voor respectievelijk coders en decoders, en tot 0 pixels in verticale richting [118, Annex A]. De publiek beschikbare implementaties waar in dit werk gebruik van gemaakt wordt[1], voldoen enkel aan deze minimale ondersteuning.

Ter referentie vermelden we dat de ligging van een contextpixel uitgedrukt wordt volgens het assenstelsel van figuur 5.2. Deze conventie wordt consequent gehanteerd doorheen de rest van het hoofdstuk. De causaliteitsvoorwaarde voor een sequentieel sjabloon herleidt zich in dit assenstelsel tot $\nu < 0$ of $\nu = 0$ en $\mu < 0$.

De standaard beschrijft ook een procedure om de optimale positie van een AT-pixel te bepalen gedurende het coderen van het beeld [118, Annex C]. Daartoe definieert het een *polariteitscoïncidentie* (*E.* polarity coincidence) als de pixels op positie (i, j) en $(i + \mu, j + \nu)$ dezelfde waarde hebben. Vervolgens telt de procedure de polariteitscoïncidenties voor elke kandidaatpositie $(\mu, 0)$. Na 2048 monsters kiest het de kandidaatpositie met het hoogste aantal coïncidenties als nieuwe AT-pixel. De wijzigingen in de AT-pixel worden expliciet gecodeerd en deze procedure wordt dan ook niet herhaald gedurende het decoderen.

De selectieprocedure is goed, maar de implementatie schaalt moeilijk voor kandidaatposities in twee dimensies en op grotere afstand. Tabel 5.1 toont het effect aan van de beperking op de bewegingsvrijheid van de AT-pixel voor de

[1] van M. Kuhn (http://www.cl.cam.ac.uk/~mgk25/download/jbigkit-1.2.tar.gz)
en van D. L. Duttweiler (ftp://nic.funet.fi/pub/graphics/misc/test-images/jbig.tar.gz)



Figuur 5.2: Conventie voor de indexering van de pixels en de oriëntatie van de referentieassen.

Tabel 5.1: Invloed van de beperking van de AT-pixel op de compressieverhouding voor een aantal halftoonbeelden. De laatste rij toont de optimale ligging $(\mu, \nu)_{opt}$ voor de AT-pixel.

AT-pixel	ro5k	ro10k	ro20k	mo10k	mo20k
vast $(2, -1)$	5.62	4.32	7.35	4.18	12.96
basis $(\|\mu\| \leq 8, \nu = 0)$	5.65	4.34	6.56	4.19	15.15
basis $(\|\mu\| \leq 16, \nu = 0)$	6.89	4.64	7.35	4.19	15.15
optimaal $(\|\mu\|, \|\nu\| \leq 32)$	7.25	5.59	9.66	4.39	15.15
positie $(\mu, \nu)_{opt}$	$(8, -8)$	$(3, -10)$	$(5, -18)$	$(0, -3)$	$(-4, 0)$

basisimplementatie. De testbeelden worden beschreven in appendix A. De prefixen "ro" en "mo" duiden respectievelijk op klassieke en stochastische halftoonbeelden. Het suffix "*n*k" betekent dat één zijde van het beeld bij benadering *n*-duizend pixels telt. Een brutekrachtmethode gaat op zoek naar de optimale kandidaatpixel binnen het causale gebied waar $|\mu|, |\nu| \leq \delta$ en $\delta = 32$. Voor de klassieke halftoonbeelden levert het vrij kiezen van de AT-pixel los van de beperking van de basisimplementatie een winst op variërend van 20% tot bijna 50%. De optimale ligging stemt overeen met de rasterperiode en is dan ook vrij ver van de oorsprong verwijderd. Voor de stochastische halftoonbeelden levert het geheel vrij kiezen van de AT-pixel weinig of geen extra winst op.

Dit toont aan dat het optimaliseren van één AT-pixel aanleiding kan geven tot een aanzienlijke winst in compressie voor halftoonbeelden. De gecodeerde stromen die hieruit volgen zijn wel compatibel met de JBIG-standaard, maar kunnen niet gedecodeerd worden aan de hand van de basisimplementatie. De

vraag dringt zich op hoe we op een effectieve manier een goede keuze kunnen maken voor het AT-pixel. Daarenboven stelt zich de vraag hoezeer de compressieverhouding nog verder zal toenemen voor een volledig vrij sjabloon en een hogere orde.

5.3.2 Autocorrelatiegebaseerd sjabloon

Voor klassieke halftoonbeelden ligt het intuïtief voor de hand om het sjabloon samen te stellen zowel uit pixels die in de onmiddellijke omgeving liggen als uit pixels die ongeveer een roosterperiode verwijderd zijn. Deze pixels hebben de interessante eigenschap dat ze in hoge mate gecorreleerd zijn met de huidige pixel. Een contextsjabloon samengesteld uit de best gecorreleerde pixels lijkt dan ook een goede keuze.

De pixels van een binair beeld kunnen slechts twee waarden aannemen. De klassieke definitie van de autocorrelatiefunctie van een beeld heeft hier weinig zin. De *polariteitscoïncidentiefunctie*

$$P(i,\, j;\, \mu,\, \nu) = \begin{cases} 1, & \text{als} \quad x_{ij} = x_{i+\mu,\, j+\nu}, \\ 0, & \text{als} \quad x_{ij} \neq x_{i+\mu,\, j+\nu}, \end{cases}$$

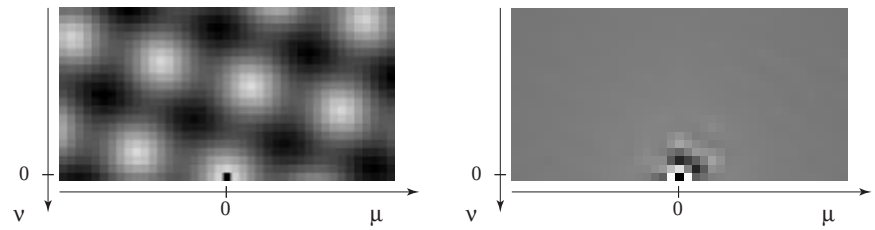
drukt uit of twee pixels x_{ij} en $x_{i+\mu,\, j+\nu}$ op relatieve afstand $(\mu,\, \nu)$ van elkaar al dan niet dezelfde waarde hebben. Daaruit volgt de *polaire autocorrelatiefunctie* van een beeld

$$A(\mu,\, \nu) = \sum_{(i,\, j) \in S} P(i,\, j;\, \mu,\, \nu) / \sum_{(i,\, j) \in S} 1,$$

waarbij de sommatie loopt over het gebied S waarvoor zowel $(i,\, j)$ als $(i + \mu,\, j + \nu)$ binnen de beeldgrenzen liggen. De q causale relatieve posities $(\mu,\, \nu)$ die aanleiding geven tot de hoogste waarde van $A(\mu,\, \nu)$ vormen samen het autocorrelatiegebaseerd contextsjabloon $T_A^q = \{(\mu_t,\, \nu_t);\ t = 1\,..\,q\}$.

Omdat deze aanpak niet schaalt voor grote beelden, voert een praktische implementatie een aantal additionele beperkingen in. Te grote waarden van μ of ν geven aanleiding tot een relatief groot aantal contextpixels $(i + \mu,\, j + \nu)$ die buiten de beeldgrenzen liggen en bijgevolg geen voorspellende waarde hebben. Daarom wordt de waarde van μ en ν beperkt tot een contextgebied dat even groot is als een achttal halftoonbasiscellen voor klassieke halftoonbeelden, wat neerkomt op een beperking $|\mu|,\, |\nu| \leq \delta$, waarbij $\delta = 20$ voor de meeste beelden, of $\delta = 32$ voor de beelden aan hoge resolutie.

Daarenboven heeft het voor grote beelden weinig zin om de autocorrelatiefunctie te berekenen over nagenoeg het volledige beeld. Daarom wordt S



Figuur 5.3: Grafische weergave van de autocorrelatiefunctie $A(\mu, \nu)$ op een representatief deel van het klassiek halftoonbeeld "ro10k" (*links*) en het stochastisch halftoonbeeld "mo10k" (*rechts*). Hoe hoger de intensiteit, hoe hoger de autocorrelatie. De oorsprong $(0, 0)$ is opzettelijk zwart gekleurd.

verder beperkt tot een representatief deel van het beeld. Voor klassieke halftoonbeelden beschouwen we een deel van het beeld als representatief indien (1) de gemiddelde intensiteit tussen 25% en 75% is, (2) de intensiteit nooit $< 5\%$ of $> 95\%$ is na toepassing van een laagdoorlaatfilter en (3) het tenminste 1000 halftooncellen bevat. Deze definitie is heel ad hoc maar blijkt vrij robuust te werken in de praktijk, onder de voorwaarde dat het oorspronkelijke grijswaardenbeeld behoorlijk stationair is. Typische waarden voor de afmetingen van het representatief beeld zijn 1024×1024 pixels, wat neerkomt op 4096 halftooncellen indien een halftooncel 256 spots bevat. Voor halftoonbeelden van heel hoge resolutie kunnen deze afmetingen oplopen tot 2048×2048.

Ter illustratie geeft figuur 5.3 de autocorrelatiefunctie $A(\mu, \nu)$ weer voor een klassiek en een stochastisch gerasterd halftoonbeeld. De autocorrelatie van het halftoonbeeld vertoont duidelijk maxima op heeltallige periodes van het basisrooster en minima op halftallige periodes. De autocorrelatie van een stochastisch halftoonbeeld vertoont een kleine "rimpel" als gevolg van het foutdiffusiealgoritme en sterft dan vlug uit.

Het bepalen van de posities waar deze autocorrelatiefunctie maximaal wordt, leidt tot een autocorrelatiegebaseerd sjabloon. Figuur 5.4 geeft deze grafisch weer voor orde $q = 20$ voor hetzelfde klassiek en stochastisch gerasterd halftoonbeeld. Het sjabloon van het klassiek halftoonbeeld bevat slechts twee contextpixels in de onmiddellijke buurt.

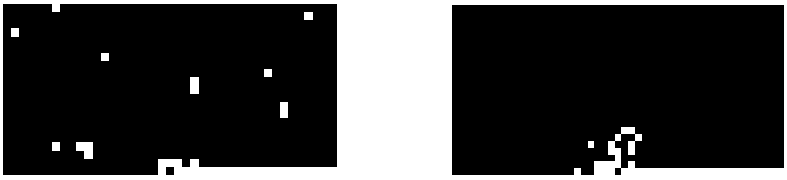
5.3.3 Suboptimaal sjabloon

Voor het evalueren van het autocorrelatiegebaseerde sjabloon stelt zich de vraag naar het sjabloon dat de hoogste compressieverhouding levert voor een gegeven beeld. Is de positie van een contextpixel beperkt tot een rechthoek van

Figuur 5.4: Grafische weergave van het autocorrelatiegebaseerd sjabloon van orde 20 voor het klassiek halftoonbeeld "ro10k" (*links*) en het stochastisch halftoonbeeld "mo10k" (*rechts*). De oosprong $(0, 0)$ ligt in het midden van de onderste lijn en is opzettelijk zwart gekleurd.

afmetingen $(2\delta + 1) \times (\delta + 1)$, dan zijn er bij benadering $(2\delta^2)!/q!(2\delta^2 - q)!$ mogelijke sjablonen van orde q. Typische waarden zijn bijvoorbeeld $\delta = 20$ en $q = 16$, wat aanleiding geeft tot een gigantisch aantal kandidaten. Bovendien kan het bepalen van het optimale sjabloon enkel via een exhaustief zoekproces. Het is dan ook niet mogelijk om binnen een redelijk tijdsinterval het optimale sjabloon te bepalen voor een realistisch beeld en een realistische contextgrootte.

Om die reden stellen we een *gretige zoektocht* (*E.* greedy search) voor die een suboptimaal sjabloon genereert. De term "suboptimaal" dient hier geïnterpreteerd te worden als "optimaal binnen realistische verwachtingen." Er wordt vertrokken van een representatief deel van het beeld van afmetingen 1024×1024 of 2048×2048, afhankelijk van de resolutie. De orde start bij $q = 1$ en het optimale sjabloon van orde $q = 1$ wordt bepaald. Vervolgens wordt dit sjabloon gefixeerd en wordt de orde met één verhoogd. Alle sjablonen bestaande uit het voorlopig gefixeerde sjabloon aangevuld met een nieuwe kandidaatpixel worden geëvalueerd. Het beste sjabloon wordt opnieuw gefixeerd en dit proces herhaalt zich totdat een vooropgestelde maximale orde q_{max} bereikt wordt of totdat de compressieverhouding opnieuw afneemt. Op deze manier reduceert de zoektocht zich tot bij benadering $2q_{max}\delta^2$ codeercycli van een beeld met beperkte afmetingen. De evaluatie van een kandidaatsjabloon kan op twee manieren gebeuren: ofwel wordt het beeld effectief gecomprimeerd, ofwel worden de statistieken opgebouwd en wordt de empirische entropie berekend. De eerste aanpak stopt bij de optimale orde voor het representatief deel van beperkte afmetingen. Maar de optimale orde voor het gehele beeld kan hoger liggen en daarom wordt het suboptimale sjabloon voor hogere ordes verder uitgebreid met de pixels die het dichtst liggen volgens de L_1-afstand. De tweede aanpak bouwt enkel statistieken op en omzeilt op deze manier de vrij intensieve entropiecoder uit het optimalisatieproces. Helaas is deze aanpak minder optimaal omdat het probleem van contextverdunning niet



Figuur 5.5: Grafische weergave van het suboptimale sjabloon van orde 20 voor het klassiek halftoonbeeld "ro10k" (*links*) en het stochastisch halftoonbeeld "mo10k" (*rechts*). De oorsprong $(0, 0)$ ligt in het midden van de onderste lijn en is opzettelijk zwart gekleurd.

in rekening gebracht wordt. Bovendien bereikt deze aanpak geen optimale orde maar blijft de efficiëntie ongehinderd toenemen bij hogere ordes. Dit heeft als gevolg dat er oneigenlijke contextpixels of *ruiscontextpixels* toegevoegd worden aan het sjabloon. In de praktijk duurt deze gretige zoektocht van een aantal uren tot een halve dag.

Figuur 5.5 illustreert het resultaat van deze gretige zoektocht voor hetzelfde klassiek en stochastisch halftoonbeeld als in figuur 5.4. De zoektocht wordt beperkt tot het gebied waarbij $\delta_{max} = 20$ is gesteld. De evaluatie van een sjabloon is gebaseerd op de empirische entropie en dit geeft aanleiding tot een aantal ruiscontextpixels in de hogere ordes. Voor het klassiek halftoonbeeld bestaat het suboptimale sjabloon uit een aantal contextpixels op gehele roosterperiodes, uit een aantal nabije contextpixels en uit een aantal intermediaire pixels. De meeste contextpixels liggen vrij ver van de oorsprong. Dit is tegengesteld aan het resultaat voor het stochastische halftoonbeeld. Hier liggen de contextpixels in de buurt van de oorsprong, al wijkt het sjabloon toch sterk af van een statisch sjabloon gebaseerd op de L_1- of de L_2-afstand.

5.3.4 Experimentele resultaten

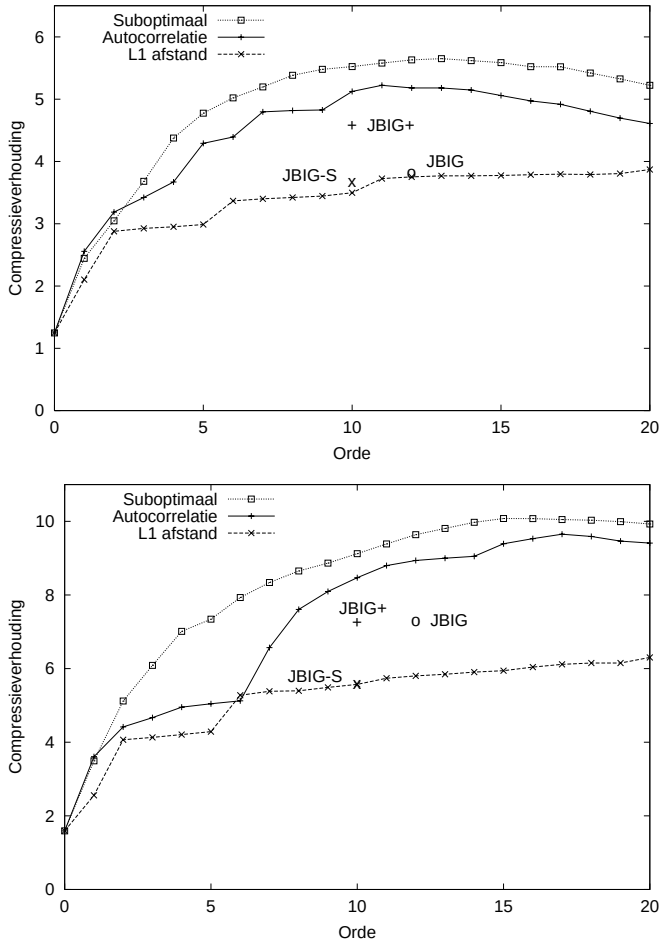
De standaard testbeelden zijn allemaal gebaseerd op dezelfde cyaancomponent van het BG "musicians-cmyk" beeld, zie appendix A. De beelden "ro1k" en "mo1k" stellen niet de gehele scène voor maar stellen elk een representatieve uitsnijding voor van respectievelijk "ro10k" en "mo10k". Ze tellen elk 1024×1024 pixels.

Figuren 5.6 en 5.7 geven de compressieverhouding weer als functie van de orde q voor de klassieke halftoonbeelden. Voor elk halftoonbeeld is er één grafiek die drie curves bevat, overeenstemmend met een sjabloon gebaseerd op de L_1-afstand, met het autocorrelatiegebaseerd sjabloon en met het suboptimaal sjabloon. De L_2-afstand als sjablooncriterium geeft aanleiding tot gelijkaar-

dige resultaten als de L_1-afstand en die resultaten ontbreken dan ook in de grafieken. Daarnaast zijn ook de volgende resultaten van JBIG opgenomen: JBIG in sequentiële modus volgens de basisimplementatie voor een coder (gemarkeerd als "$\times$", JBIG-S), JBIG in sequentiële modus volgens de uitgebreide implementatie met volledig vrije AT-pixel (gemarkeerd als "$+$", JBIG+) en JBIG in multiresolutiemodus (gemarkeerd als "$\circ$", JBIG). Het autocorrelatiegebaseerd sjabloon is berekend op een representatief deel van 1024×1024 pixels met $\delta_{max} = 20$. Het suboptimale sjabloon is berekend op een representatief deel van 2048×2048 pixels met $\delta = 24$. De procedure om dit representatief deel te bepalen is eerder ad hoc maar levert voor typische testbeelden meestal een consistent resultaat op. De evaluatie van een sjabloon is gebaseerd op de reële compressieverhouding en niet op de schatting van de empirische entropie. Merk op dat de optimale orde voor een beeld van die afmetingen varieert van 14 tot 19 en dat de zoektocht stopt bij die orde. Een artificiële uitbreiding van het optimale sjabloon met nabijgelegen pixels leidt tot een sjabloon voor hogere ordes.

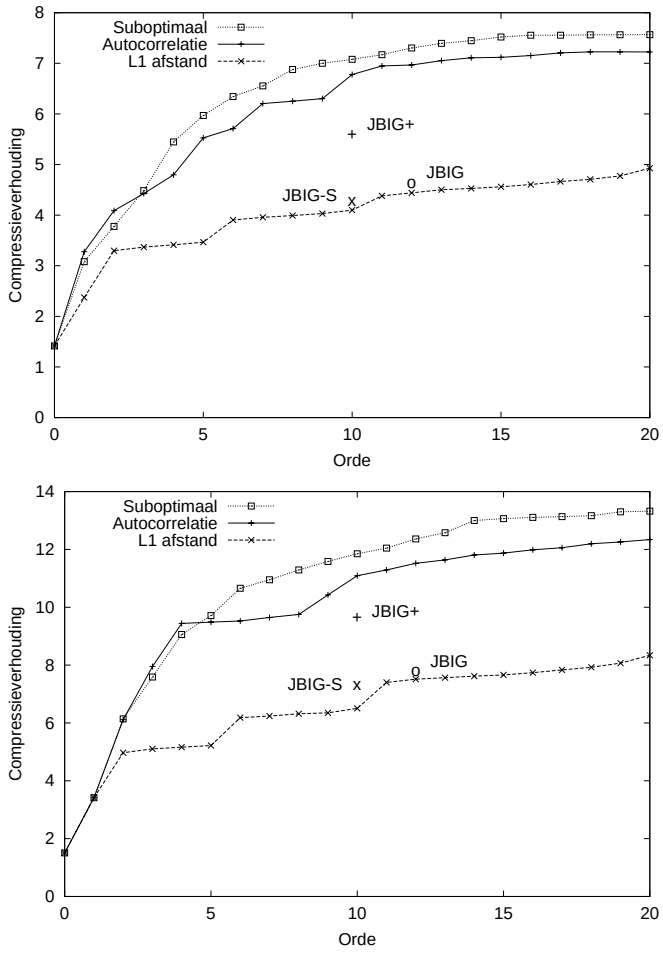
De resultaten van figuren 5.6 en 5.7 laten toe een aantal conclusies te trekken voor klassieke halftoonbeelden. Het beeld "ro1k" is niet representatief wegens de beperkte afmetingen. Toch illustreert het duidelijk het kwalitatief gedrag rond de optimale orde. De compressieverhouding neemt initieel heel snel toe, vlakt vervolgens af ter hoogte van het maximum bij de optimale orde en neemt daarna gestaag af. Deze afname is een direct gevolg van het probleem van contextverdunning. Het beeld "ro5k" heeft vrij unieke rasterparameters en is hierdoor evenmin representatief. De rasterperiode is $(8, 8)$ en hierdoor zijn de halftooncelgrenzen gealigneerd met de bytegrenzen van de binaire voorstelling. Algemene technieken en multiresolutietechnieken zoals JBIG in multiresolutiemodus doen het verrassend goed. Toch kan er nog aanzienlijke winst geboekt worden door een semi-adaptief sjabloon te gebruiken in sequentiële modus.

De andere twee beelden, "ro10k" en "ro20k" zijn wel representatief. De compressieverhouding op basis van een autocorrelatiegebaseerd sjabloon is aanzienlijk hoger dan dat van een statisch sjabloon gebaseerd op de L_1-afstand. Anders gesteld kan een gelijkaardig resultaat bekomen worden aan een veel lagere orde en bijgevolg ook aan een veel lager geheugengebruik. De compressieverhouding blijft oplopen tot circa orde 16 tot 20, maar de grootste winst wordt al bij lage orde bekomen. JBIG in sequentiële mode met geoptimaliseerd AT-pixel ("JBIG+" in de figuur) levert ongeveer de helft van de winst op die bekomen kan worden aan de hand van een volledig semi-adaptief sjabloon. En de resultaten met het autocorrelatiegebaseerd sjabloon komen dicht in de buurt van die op basis van het suboptimaal sjabloon.



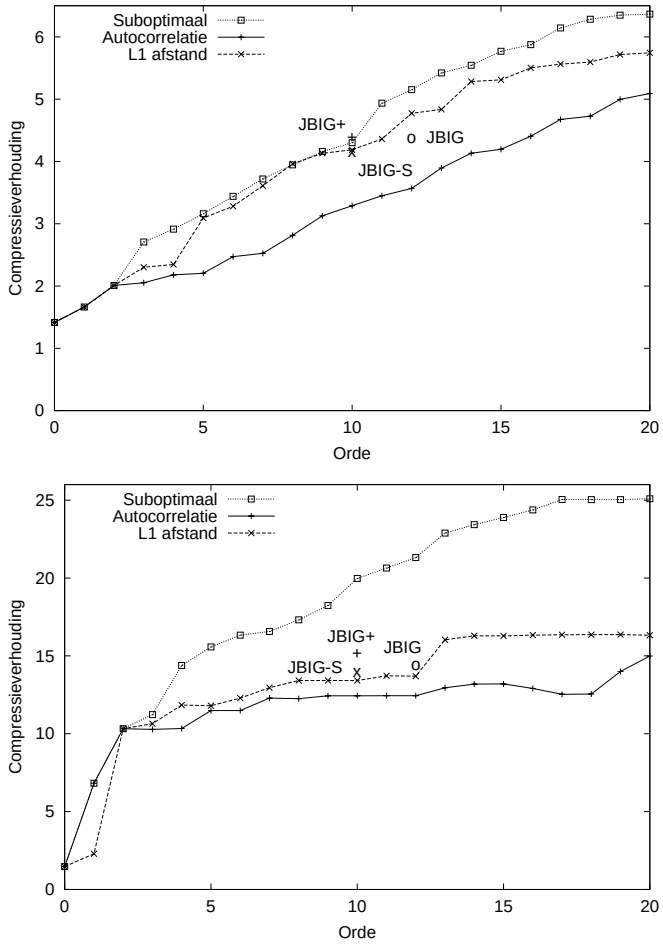
Figuur 5.6: Grafiek van de compressieverhouding als functie van de orde voor de kleine klassieke halftoonbeelden "ro1k" (*boven*) en "ro5k" (*onder*). Er zijn drie curves overeenstemmend met een sjabloon gebaseerd op de L_1-afstand, op de autocorrelatie en op een gretige zoektocht. De volgende resultaten van JBIG zijn eveneens opgenomen: sequentiële modus volgens de basisimplementatie ("$\times$" JBIG-S), sequentiële modus met geheel vrij AT-pixel ("$+$" JBIG+) en multiresolutiemodus ("$\circ$" JBIG).

Figuur 5.8 geeft gelijkaardige grafieken weer voor de stochastische halftoonbeelden "mo10k" en "mo20k". De resultaten zien er helemaal anders uit. De compressieverhouding van JBIG kan aanzienlijk verhoogd worden door een hogere orde te kiezen en een semi-adaptief sjabloon te gebruiken. Helaas levert een sjabloon gebaseerd op autocorrelatie slechtere resultaten op dan



Figuur 5.7: Grafiek van de compressieverhouding als functie van de orde voor de grote klassieke halftoonbeelden “ro10k” (*boven*) en “ro20k” (*onder*). De legende is dezelfde als bij figuur 5.6.

een sjabloon gebaseerd op de L_1-afstand. Het verloop van de compressieverhouding als functie van de orde vertoont een meer lineair gedrag dan bij de klassieke halftoonbeelden. Voor het beeld “mo20k” geeft het gebruik van het suboptimaal sjabloon aanleiding tot een aanzienlijke verbetering. De precieze verklaring is vermoedelijk te zoeken in de werking van het halftoonalgoritme dat doorgaans halftoondots van 4×4 pixels genereert. Helaas zijn de details van dit proces onvoldoende gekend en vermoeden we dat de resultaten sterk afhankelijk zijn van de details van het halftoonproces.



Figuur 5.8: Grafiek van de compressieverhouding als functie van de orde voor de stochastische halftoonbeelden “mo10k” (*boven*) en “mo20k” (*onder*). De legende is dezelfde als bij figuur 5.6.

Tabel 5.2 vat deze resultaten numeriek samen voor een aantal sjablonen met diverse karakteristieken. Daarenboven geeft de tabel ook het optimale resultaat weer dat haalbaar is in sequentiële modus met semi-adaptief sjabloon binnen de JBIG2-standaard. Voor de klassieke halftoonbeelden geeft het autocorrelatiegebaseerd sjabloon aanleiding tot een winst van ongeveer 50% ten opzichte van JBIG-S in de basisimplementatie en ongeveer 20% ten opzichte van JBIG1 in de uitgebreide implementatie, dit voor dezelfde orde $q = 10$. Voor een suboptimaal sjabloon loopt deze winst op tot respectievelijk meer

Tabel 5.2: Compressieverhouding van de referentiehalftoonbeelden volgens een aantal sjablonen van verschillende orde. De notatie "xF-yA" staat voor x vaste contextpixels en y AT-pixels. Technieken compatibel met de basisimplementatie van JBIG zijn gemarkeerd met "*". De notatie "nL" wijst er op dat het vaste sjabloon zich beperkt tot n lijnen. De resultaten gemarkeerd met "†" zijn gebaseerd op een artificieel uitgebreid sjabloon. De laatste lijn geeft de optimale orde q'_{opt} weer voor het representatief deel van 2048×2048 pixels.

sjabloon	ro5k	ro10k	ro20k	mo10k	mo20k
JBIG* multiresolutie	7.31	4.65	7.83	4.39	14.47
L_1-afstand 10F	5.57	4.10	6.50	4.19	13.42
L_2-afstand 10F	5.62	4.32	7.35	4.18	12.96
JBIG-S* 9F-1A*	5.60	4.30	7.32	4.16	13.98
JBIG+ 9F-1A (2L)	7.05	5.47	9.15	4.34	16.75
JBIG+ 9F-1A (3L)	7.14	5.59	9.34	4.40	15.15
JBIG2 12F-1A	7.45	5.96	10.22	5.04	16.05
JBIG2 12F-4A	9.15	7.21	12.11	5.59	18.80
autocorrelatie 6A	5.12	5.71	9.53	2.47	11.49
autocorrelatie 10A	8.47	6.78	11.09	3.29	12.43
autocorrelatie 12A	8.94	6.97	11.52	3.57	12.44
autocorrelatie 16A	9.53	7.15	11.99	4.40	12.91
autocorrelatie 20A	9.41	7.22	12.34	5.09	14.99
suboptimaal 6A	7.93	6.34	10.66	3.44	16.33
suboptimaal 10A	9.12	7.08	11.85	4.30	19.97
suboptimaal 12A	9.64	7.30	12.36	5.15	21.32
suboptimaal 16A	$10.07^\dagger$	$7.56^\dagger$	$13.11^\dagger$	5.87	24.37
suboptimaal 20A	$9.93^\dagger$	$7.57^\dagger$	$13.32^\dagger$	$6.36^\dagger$	$25.09^\dagger$
orde (q'_{opt})	(15)	(15)	(14)	(19)	(17)

dan 60% en meer dan 25%. De compessieverhouding met het moeilijk te berekenen suboptimaal sjabloon is slechts 4% tot 8% beter dan met het eenvoudig te bepalen autocorrelatiegebaseerd sjabloon. Het verhogen van de orde van $q = 10$ tot $q = 20$ geeft aanleiding tot een additionele winst van 6% tot 12% waarbij hetzelfde sjabloon gebruikt wordt.

Voor de stochastische halftoonbeelden zien de resultaten er helemaal anders uit. Voor orde $q = 10$ geeft het autocorrelatiegebaseerd sjabloon aanleiding tot een verlies ten opzichte van JBIG-S in sequentiële modus. Het suboptimale sjabloon geeft wel aanleiding tot een winst die kan oplopen tot 40%. Het verhogen van de orde van $q = 10$ tot $q = 20$ leidt tot een additionele

Tabel 5.3: Afmetingen van de halftoonbeelden voor en na compressie, uitgedrukt in 10^6 byte. Voor elke techniek is de optimale orde gekozen. De notatie is volgens dezelfde conventie als in tabel 5.2.

techniek	ro5k	ro10k	ro20k	mo10k	mo20k
—	3.11	12.86	51.43	12.86	51.43
JBIG-S*	0.56	2.99	7.02	3.10	3.68
JBIG+ 9F-1A	0.44	2.30	5.51	2.92	3.07
autocorrelatie	0.32	1.78	4.17	2.53	3.43
suboptimaal	0.31	1.70	3.86	2.02	2.05

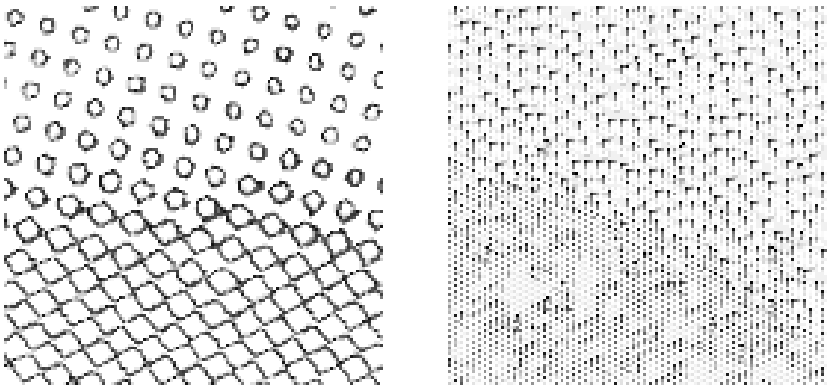
winst van 25% tot 50%. In tegenstelling tot klassieke halftoonbeelden zijn de resultaten hier heel gevoelig voor de exacte details van het halftoonproces.

In het kader van de discussie van verliesloze compressie van halftoonbeelden versus contone beelden is het interessant om de grootte van de gecomprimeerde beelden te vergelijken voor en na het halftoonproces. De beste techniek voor verliesloze compressie van contone beelden is CALIC en die genereert een bestand van 2.08×10^6 byte voor de "musicians-c" component. De afmetingen van de gecomprimeerde halftoonbestanden met de optimale parameters zijn samengevat in tabel 5.3. Gebruiken we de lengte van een gecomprimeerd bestand als maat voor de informatie van de voorstelling, dan heeft het halftoonproces vrij weinig impact op de hoeveelheid informatie van een beeld. Enkel voor het klassieke halftoonbeeld "ro20k" is er een significante toename merkbaar. Dit kan ten dele verklaard worden door het feit dat de randen van de halftoondots heel moeilijk te voorspellen zijn aan hoge resolutie.

Figuur 5.9 is een artificiële afbeelding van het bitverbruik voor iedere pixel, ingezoomd op eenzelfde stuk van 128×128 pixels voor het klassiek halftoonbeeld "ro10k" en het stochastisch halftoonbeeld "mo10k". Bij het klassieke halftoonbeeld is vooral de rand van elke dot moeilijk te voorspellen. De halftooncentra en de ruimte tussen de dots verbruiken heel weinig bits in de gecomprimeerde stroom. De context slaagt erin de spatiale fase van de halftoonpixel te voorspellen. Bij het stochastische halftoonbeeld gaat de informatie vooral naar het precieze startpunt van de halftoondot. De context probeert het lokale intensiteitsniveau en de lokale intensiteitsvariatie te schatten.

5.3.5 Invloed van de halftoonparameters

Het aantal halftoonbeelden dat tot hiertoe aan bod kwam is eerder beperkt. Hierna onderzoeken we de invloed van andere contone beelden, van een aantal



Figuur 5.9: Bitverbruik van eenzelfde ingezoomd stuk (128 $\times$ 128 pixels) van het klassieke halftoonbeeld "ro10k" (*links*) en het stochastisch halftoonbeeld "mo10k" (*rechts*). Hoe donkerder de pixel, hoe onwaarschijnlijker de waarde en hoe meer bits er nodig zijn om de waarde te coderen.

externe factoren die een rol kunnen spelen bij het ontstaansproces van het halftoonbeeld en van een aantal vrijheidsgraden voor het klassieke halftoonproces zoals resolutie, dottype en rasterhoek. Appendix A beschrijft de nieuwe testbeelden en hun parameters. Ook hier gaat de meeste aandacht naar klassieke halftoonbeelden wegens hun dominerende aanwezigheid in de drukvoorbereidingsindustrie.

JBIG Stockholm halftoonbeelden

De JBIG Stockholm halftoontestbeelden zijn de enige halftoonbeelden waarvoor resultaten gepubliceerd zijn met andere technieken [6, 146, 147]. Tabel 5.4 geeft de resultaten weer van een aantal standaardtechnieken, van de voorgestelde methode, evenals van een recent gepubliceerde techniek. De klassieke halftoonbeelden "s04{a-d}-b" zijn klassieke halftoonbeelden die dezelfde scène voorstellen, maar die gerasterd zijn volgens verschillende halftoonparameters. De periode van het niet geroteerde raster is geheel en varieert van 3 tot 8 pixels. Het stochastisch halftoonbeeld "s09a-b" is gegenereerd door foutdiffusie maar de resolutie is te laag om representatief te zijn.

De optimalisatieparameters zijn dezelfde als voor de vorige resultaten: de autocorrelatie is gebaseerd op een gebied van 1024 $\times$ 1024 pixels met $\delta = 20$ en de gretige zoektocht gebeurt op een gebied van 2048 $\times$ 2048 pixels met $\delta = 24$. De resultaten zijn kwalitatief gelijkaardig aan die van de BG halftoonbeelden. Voor de klassieke halftoonbeelden komen de resultaten van het

Tabel 5.4: Compressieverhouding van de halftoonbeelden van de JBIG Stockholm set volgens een aantal sjablonen van verschillende orde. De notatie is dezelfde als in tabel 5.2. Ter referentie zijn de resultaten van een gelijkaardige maar meer geavanceerde techniek opgenomen [147].

sjabloon	s04a	s04b	s04c	s04d	s09
JBIG* multiresolutie	9.29	7.18	11.98	7.17	1.34
L_1-afstand 10F	7.81	6.25	6.64	11.82	1.74
L_2-afstand 10F	6.13	5.07	6.08	6.71	1.77
JBIG-S* 9F-1A*	10.34	4.99	11.21	11.77	1.76
JBIG+ 9F-1A (2L)	14.28	14.09	18.02	17.95	1.73
JBIG+ 9F-1A (3L)	13.61	13.99	15.06	16.45	1.80
JBIG2 12F-1A	14.78	15.18	16.11	19.47	1.82
JBIG2 12F-4A	18.92	20.33	22.21	22.48	1.86
autocorrelatie 6A	14.08	16.01	18.69	18.91	1.37
autocorrelatie 10A	15.69	16.98	19.82	20.11	1.40
autocorrelatie 16A	16.45	17.37	20.36	20.26	1.30
suboptimaal 6A	15.34	16.63	19.50	19.76	1.47
suboptimaal 10A	17.32	18.86	21.25	21.20	1.74
suboptimaal 16A	17.65	21.36	21.51	21.37	1.92
orde (q'_{opt})	(14)	(16)	(14)	(13)	(16)
"free template" [147]	19.08	20.77	22.40	22.70	1.90

autocorrelatiegebaseerde sjabloon dicht in de buurt van die van het suboptimale sjabloon. Voor het stochastische halftoonbeeld is een autocorrelatiegebaseerd sjabloon geen goede keuze. De laatste lijn geeft de resultaten weer van een externe techniek "free template" [145, 147]. Deze is eveneens gebaseerd op een gretige zoektocht naar een suboptimaal semi-adaptief sjabloon, maar doet andere toegevingen. De bekomen compressieverhouding is gelijkaardig. Een aantal verrassende resultaten — zo is "JBIG2 12F-4A" soms beter dan "suboptimaal, 16A" — is te verklaren door de gretigheid van de zoektocht, het gebruik van een niet-optimale orde en door het niet-stationair karakter van de beeldintensiteit.

Invloed van belichten en scannen

Onder normale omstandigheden zijn de halftoonbeelden digitaal gegenereerd door het rasteralgoritme toe te passen op een contoon beeld. Dit deterministisch digitaal proces bepaalt grotendeels de invloed van het rasteren op de hoe-

veelheid ruis in het halftoonbeeld. Het kan ook voorkomen dat een halftoonbeeld ontstaat uit een analoog bemonsteringsproces, zoals wanneer het halftoonbeeld belicht is op film en opnieuw ingescand wordt.

Om dit te onderzoeken, hebben we de BG halftoonbeelden belicht op film en vervolgens opnieuw ingescand aan de oorspronkelijke halftoonresolutie. Tabel 5.5 vergelijkt de resultaten op de oorspronkelijke digitaal gegenereerde halftoonbeelden met de belichte en opnieuw gedigitaliseerde halftoonbeelden. Voor de beelden aan gewone resolutie (680 en 1270 dpi) is de invloed van deze extra processtap beperkt. Het effect is wisselend in het voordeel en het nadeel en beperkt zich tot circa 3%. Voor het klassiek halftoonbeeld aan 2540 dpi is er een winst die oploopt tot ongeveer 10%. Het grootste verschil wordt bereikt voor het stochastische halftoonbeeld aan 2540 dpi. De comprimeerbaarheid neemt af met circa 40–65%. Na nauwkeurige inspectie van de halftoonbeelden blijkt een mogelijke verklaring te liggen in het feit dat de fysische resolutie van de invoer- en uitvoertoestellen bereikt wordt. Voor een klassiek halftoonbeeld resulteert dit in dots van een meer regelmatige vorm, wat een positief gevolg heeft voor de comprimeerbaarheid. Voor het onderzochte stochastisch halftoonbeeld is het effect tegengesteld, want de nagenoeg rigide vierkantdots van 4×4 pixels krijgen een onvoorspelbare rand.

Daarnaast tonen de resultaten aan dat de voorgestelde aanpak vrij ongevoelig is voor kleine rotaties van het halftoonrooster en voor kleine afwijkingen in de periodiciteit van het rooster. Dit experiment is heel beperkt en de resultaten dienen dan ook met de nodige voorzichtigheid geïnterpreteerd te worden.

Invloed van de resolutie

Het is te verwachten dat de compressieverhouding afhankelijk is van de halftoonresolutie en de lijnfrequentie. Daarnaast wensen we na te gaan of de resultaten van de voorgestelde methoden consequent bereikt worden voor andere parameters voor het klassiek halftoonproces.

De resultaten voorgesteld in deze en volgende paragrafen zijn bekomen op basis van een reeks halftoonbeelden gegenereerd met behulp van het beeldverwerkingsprogramma ADOBE PHOTOSHOP. Alle halftoonbeelden stellen dezelfde contone component “musicians-c” voor. De halftoonparameters resolutie, dotvorm en resolutie variëren rond de respectieve centrale waarden “1270 dpi”, “rond” en “15°”. Tabel A.4 beschrijft de halftoonparameters van elk testbeeld in detail. Bij nauwkeurige inspectie van de halftoonbeelden komen enkele lichte anomalieën naar voor, zoals lijnvormige artefacten. Deze zijn toe te schrijven aan het halftoonalgoritme, dat van een lagere kwaliteit is dan van professionele drukvoorbereidingssystemen zoals die van Barco Grap-

Tabel 5.5: Invloed van belichten en inscannen op de compressieverhouding voor de standaard referentieset van halftoonbeelden.

sjabloon	ro5k	ro10k	ro20k	mo10k	mo20k
I - Digitaal gegenereerd					
L_1-afstand 10F	5.57	4.10	6.50	4.19	13.42
JBIG-S* 9F-1A*	5.60	4.30	7.32	4.16	13.98
JBIG+ 9F-1A	7.14	5.59	9.34	4.40	15.15
JBIG2 12F-4A	9.15	7.21	12.11	5.59	18.80
autocorrelatie 16A	9.53	7.15	11.99	4.40	12.91
suboptimaal 16A	10.07†	7.56†	13.11†	5.87	24.37
II - Belicht en ingescand					
L_1-afstand 10F	5.80	4.32	7.29	4.11	8.03
JBIG-S* 9F-1A*	5.85	4.50	8.05	4.09	7.79
JBIG+ 9F-1A	7.28	5.91	9.99	4.31	7.80
JBIG2 12F-4A	9.18	7.39	12.81	5.53	8.57
autocorrelatie 16A	9.54	7.45	13.28	4.32	7.83
suboptimaal 16A	9.91†	7.82†	14.11†	5.97	8.88

hics NV (nu Esko-Graphics NV).

Omdat de nadruk ligt op het onderzoeken van het gedrag van de compressieverhouding als functie van de halftoonparameter, stellen de resultaten niet altijd het best mogelijke resultaat voor dat mogelijk is. Zo is de optimale orde nagenoeg altijd hoger dan 16, maar het suboptimaal sjabloon is beperkt tot 16 om een faire vergelijking met andere technieken mogelijk te maken. Voor het autocorrelatiegebaseerd sjabloon is δ resolutieafhankelijk (20, 24, 28 en 32 voor respectievelijk een resolutie van $300-900$, $1200-1500$, $1800-2400$ en $3000-3600$) en bevat het representatief gebied 2048×2048 pixels. Het suboptimaal sjabloon wordt bepaald op basis van hetzelfde gebied met als consistente beperking $\delta = 24$.

Tabel 5.6 geeft de resultaten weer als functie van de resolutie. Elke kolom stemt overeen met een typische combinatie van halftoonresolutie en lijnfrequentie. Voor het suboptimaal sjabloon schaalt de compressieverhouding bij ruwe benadering lineair als functie van de lijnfrequentie, maar loopt ze veel trager dan lineair op als functie van de halftoonresolutie. Dit betekent dat de binaire beeldinformatie bij toenemende resolutie als ruis beschouwd wordt door het contextmodel. Voor elke resolutie is het relatief gedrag van de onderlinge technieken in overeenstemming met het gedrag voor de BG halftoonbeelden.

Tabel 5.6: Invloed van de halftoonresolutie op de compressieverhouding voor eenzelfde contoon beeld. De kolom met als titellijnen r en l correspondeert met het testbeeld "mc-h-r-rd-l-15". Hierbij stelt r de halftoonresolutie voor en l de lijnfrequentie overeenkomstig tabel A.4.

sjabloon	300	600	900	1200	1500	1800	2400	3000	3600
	75	90	100	110	115	120	130	140	150
JBIG* multires.	2.46	5.34	3.99	5.03	5.67	6.84	9.31	9.61	11.47
L_1-afstand 10F	2.88	3.16	3.76	4.36	4.81	5.27	7.64	8.09	9.65
JBIG-S* 9F-1A*	3.03	3.36	3.82	4.54	5.07	5.57	8.38	8.78	10.59
JBIG+ 9F-1A	4.90	5.52	6.58	7.20	7.70	7.40	10.38	11.38	13.09
JBIG2 12F-4A	7.32	8.63	9.47	10.22	10.88	10.97	13.59	14.33	16.10
autocorr. 16A	6.23	7.35	9.61	10.47	11.52	11.94	14.17	15.94	16.92
subopt. 16A	7.30	9.19	10.46	11.29	12.06	12.90	14.70	17.12	18.24

Het autocorrelatiegebaseerde sjabloon scoort 5% tot 10% minder goed dan het suboptimale sjabloon.

Invloed van de dotvorm

De dotvorm van een halftoonbeeld is eveneens een belangrijke vrijheidsgraad voor het halftoonproces. Tabel 5.7 geeft de resultaten weer voor de dotvormen "rond" (rd), "kruis" (cr), "diamant" (dm), "ellips" (el), "lijn" (ln) en "vierkant" (sq). Met uitzondering van het lijnraster is de afhankelijkheid van de dotvorm beperkt. Het halftoonbeeld gegenereerd aan de hand van het lijnraster bevat significant meer redundantie. Voor elke dotvorm is het relatief gedrag van de onderlinge technieken in overeenstemming met het gedrag voor de BG halftoonbeelden.

Invloed van de rasterhoek

Tenslotte is ook de rasterhoek een belangrijke vrijheidsgraad voor het halftoonproces. Tabel 5.8 geeft de resultaten weer voor een rasterhoek variërend van 0° tot 45° in stappen van 7.5°. Om symmetrieredenen zijn geen grotere hoeken in het experiment opgenomen. Voor de semi-adaptieve sjablonen zijn de resultaten nagenoeg hoekonafhankelijk. Dit is in tegenstelling tot de multiresolutietechniek en de statische sjablonen waar een hoek van 0° of 45° aanleiding geeft tot een veel hogere compressieverhouding. Ook hier is het relatief

Tabel 5.7: Invloed van de halftoonvorm op de compressieverhouding voor eenzelfde contoon beeld. De kolom met als titellijn x correspondeert met het testbeeld "mc-h-1270-x-110-15". Hierbij beschrijft x het dottype overeenkomstig tabel A.4.

sjabloon	rd	cr	dm	el	ln	sq
JBIG* multires.	5.05	4.38	4.87	5.09	5.65	3.75
L_1-afstand 10F	4.38	4.30	4.08	4.34	7.41	4.22
JBIG-S* 9F-1A*	4.55	4.27	4.45	4.54	7.34	4.32
JBIG+ 9F-1A	7.26	7.37	6.96	7.25	9.58	7.33
JBIG2 12F-4A	10.35	10.16	10.01	10.40	13.22	10.20
autocorr. 16A	10.61	10.33	10.21	10.32	13.34	10.16
subopt.	11.44	11.35	11.19	11.62	13.90	11.36

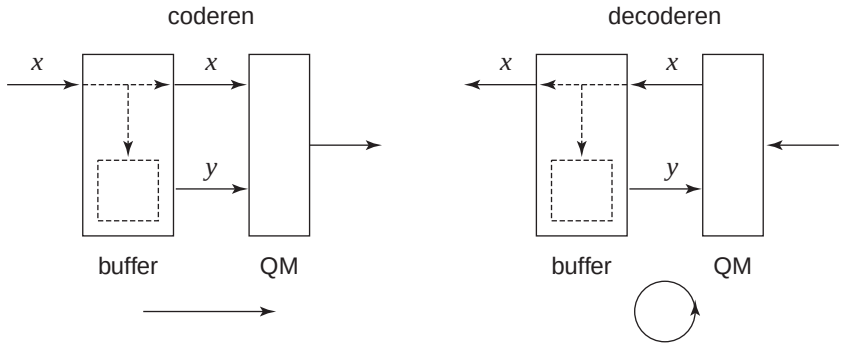
Tabel 5.8: Invloed van de rasterhoek op de compressieverhouding voor eenzelfde contoon beeld. De kolom met als titellijn "α" correspondeert met het testbeeld "mc-h-1270-rd-110-α". Hierbij beschrijft α de rasterhoek overeenkomstig tabel A.4.

sjabloon	0°	7.5°	15°	22.5°	30°	37.5°	45°
JBIG* multires.	8.87	4.70	5.05	6.99	6.12	5.01	8.89
L_1-afstand 10F	6.74	4.37	4.38	5.09	4.48	4.19	6.03
JBIG-S* 9F-1A*	6.82	4.47	4.55	5.42	4.66	4.38	6.08
JBIG+ 9F-1A	8.75	7.29	7.26	7.64	7.34	7.16	8.28
JBIG2 12F-4A	10.77	10.09	10.35	10.55	10.32	10.24	10.60
autocorr. 16A	10.27	10.47	10.61	10.86	10.82	10.70	10.46
subopt.	11.72	11.32	11.44	11.70	11.65	11.54	11.94

gedrag van de onderlinge technieken voor een gegeven rasterhoek in overeenstemming met het eerder waargenomen gedrag.

5.4 Implementatie

De bekomen winst in compressie is te danken aan het semi-adaptief sjabloon dat volgt uit een schatting van de autocorrelatie. Dit semi-adaptief sjabloon resulteert in enerzijds een additionele stap bij het coderen van het beeld en anderzijds een complexere contextvorming. De overige bewerkingen, zoals het eigenlijke contextmodel en de aritmetische coder, zijn verwerkt in de QM-coder en blijven ongewijzigd. Het is dan ook de vraag welke invloed dit heeft



Figuur 5.10: Het datapad voor het decoderen is veel kritischer dan voor het coderen, waarbij x de pixelwaarde en y de context voorstelt.

op de uiteindelijke verwerkingssnelheid. Voor realistische toepassingen kunnen de snelheidsvereisten verschillen voor coderen versus decoderen.

Het vooraf bepalen van het sjabloon heeft een negatief effect op het eventuele pijplijnen van het algoritme als bouwblok in een groter proces. Maar zelfs zonder deze extra bewerking komt de mogelijkheid van een sjabloon dat een groot aantal lijnen terugkijkt de buffergrootte niet ten goede. Dit vormt een tweede nadeel voor het extern pijplijnen van zowel het codeer- als het decodeeralgoritme. Wat het intern pijplijnen betreft is er een groot verschil tussen het coderen en het decoderen. Omdat tijdens het coderen alle pixels gekend zijn, kunnen alle contexten bepaald worden los van de entropiecoder. Het kritisch datapad vormt geen beperking. Dit in tegenstelling tot het decoderen, waar de contextpixels volgen uit een eerdere cyclus van de entropiedecoder. Het kritisch datapad kan hier bijzonder kort zijn, waarbij de sjabloonpositie $(\mu, \nu) = (-1, 0)$ het meest kritische pad veroorzaakt. Figuur 5.10 illustreert deze intrinsiek aanwezige data-afhankelijkheid tussen contextvorming en entropiecoder.

De QM-coder is geoctrooieerd door IBM, maar er zijn meerdere implementaties publiek beschikbaar[2]. De QM-coder is zo ontworpen dat de implementatie ervan vrij is van vermenigvuldigingen en dat ze slechts één byte geheugen per mogelijke context gebruikt. Voor sjablooncodering van orde q zijn dan ook 2^q bytes nodig voor het eigenlijke contextmodel. Merk op dat het semi-adaptief maken van het sjabloon de geheugenkost niet wezenlijk doet toenemen. Hierna gaan we niet verder in op de implementatie van de QM-coder,

[2]van M. Kuhn (http://www.cl.cam.ac.uk/~mgk25/download/jbigkit-1.2.tar.gz)
en van D. L. Duttweiler (ftp://nic.funet.fi/pub/graphics/misc/test-images/jbig.tar.gz)

maar besteden we vooral aandacht aan de autocorrelatiegebaseerde sjabloongeneratie enerzijds en de optimalisatie van de contextvorming anderzijds. Voor deze laatste beschouwen we zowel een software- als hardware-implementatie.

5.4.1 Schatting van de autocorrelatie

De autocorrelatieschatting baseert zich op statistieken van een groot aantal bitvergelijkingen. Elke bitvergelijking stemt overeen met één evaluatie van de polariteitscoïncidentiefunctie. Omdat deze stap enkel nodig is voor het coderen, beperken we ons tot een geoptimaliseerde implementatie in software. Merk op dat de optimale keuze van het sjabloon hoofdzakelijk bepaald wordt door de rasterparameters. Voor veel toepassingen zou dan ook een beeldonafhankelijk voorgedefinieerd sjabloon kunnen gebruikt worden dat uitsluitend afhangt van het rasteralgoritme en haar instellingen.

Voor spatiaal uitgestrekte kandidaatsjablonen kan het aantal bitvergelijkingen sterk oplopen. Bevat het representatief gedeelte van het beeld waarvoor de autocorrelatie berekend wordt $m' \times n'$ pixels, en wordt de kandidaatpositie van een contextpixel beperkt door $-\delta \leq \mu \leq \delta$ en $\delta \leq \nu \leq 0$, dan moeten bij benadering $m'n'(2\delta+1)(\delta+1)$ bitvergelijkingen verwerkt worden. Voor typische waarden $m' = n' = 1024$ en $\delta = 20$ resulteert dit in 0.9×10^9 bitvergelijkingen. Voor beelden aan hoge resolutie is dit onvoldoende en geeft $m' = n' = 2048$ en $\delta = 32$ aanleiding tot 9.0×10^9 bitvergelijkingen.

Dit kan op meerdere manieren geïmplementeerd worden, maar elke implementatie bevat minstens vier geneste lussen (respectievelijk voor i, j, μ en ν), een bitvergelijkingsfunctie en een sommatie. De volgende combinatie van deze bouwblokken gaf empirisch aanleiding tot de hoogste verwerkingssnelheid. De buitenste lus telt de lijnen $0 \leq j \leq n'$, de middelste lussen tellen de kandidaatposities (μ, ν) en de binnenste lus telt de pixels $1 \leq i \leq m'$ in blokken van grootte $k = 32$. Elk blok van $k = 32$ bits bevat twee woorden van $k' = 16$ bits. Op het huidige woord a en het referentiewoord b op afstand (μ, ν) wordt de bitsgewijze XOR-functie toegepast, wat resulteert in het verschilwoord $d = a \wedge b$. De polariteitscoïncidenties worden vervolgens geteld aan de hand van een opzoektabel $c_1(d)$, die voor elk woord d van lengte $k' = 16$ het aantal '1'-bits geeft in d. Om de geheugentoegang te optimaliseren wordt tot slot gebruik gemaakt van een schuifbuffer B die $(2\delta+1)(\delta+1)$ lijnen bevat. Voor een gegeven waarde j en voor elke mogelijke kandidaatpositie (μ, ν), bevat $B(\mu, \nu)$ de lijn $j + \nu$ verschoven over een afstand μ naar links. Dit schuifbuffer wordt opnieuw geïnitialiseerd telkens j een nieuwe waarde aanneemt. De schuifbuffer wijzigt niet gedurende de lussen voor (μ, ν) en j. De pseudocode in figuur 5.11 vat de structuur samen van de geoptimaliseerde

```
-- schatting autocorrelatie
for lijn j = 1 .. n' do:
  initialiseer B:
    B(μ, ν) = (lijn j + ν) ≪ μ voor elke (μ, ν)
  for lijnverschil ν = −δ .. 0 do:
    for kolomverschil μ = −δ .. δ do:
      for woord i' = 1 .. m'/k' do:
        a = woord i' in lijn j
        b = woord i' in lijn B(μ, ν)
        A(μ, ν) += c1(a ∧ b)
```

Figuur 5.11: Pseudocode voor een snelle schatting van de autocorrelatie.

implementatie.

Deze geoptimaliseerde implementatie is het resultaat van het experimenteel testen van een groot aantal alternatieven. Tijdens deze testen is gebleken dat de optimale woordgrootte k' en de hiermee geassocieerde adressering van c_1 afhankelijk is van het platform en de grootte van de geheugencaches. Deze platformafhankelijke verschillen zijn evenwel beperkt.

5.4.2 Contextvorming — software

Het semi-adaptief contextsjabloon heeft twee gevolgen voor de implementatie van de contextvorming. Ten eerste is de waarde van het sjabloon niet gekend tijdens het compileren en ten tweede kan het sjabloon zich spatiaal heel ver uitstrekken. Als gevolg hiervan wordt de contextberekening een operatie die vrij intensieve en moeilijk voorspelbare geheugentoegang nodig heeft. Niettegenstaande de contextvorming functioneel identiek is voor coderen en decoderen, geeft het kritische datapad voor decompressie toch aanleiding tot een fundamenteel verschil voor het optimaliseren van beide implementaties.

In software stellen we geen expliciete beperkingen aan de orde en de spatiale uitgestrektheid van het sjabloon. De voorgestelde implementatie heeft als hoofddoel de verwerkingssnelheid te maximaliseren terwijl de geheugenvereisten zo beperkt mogelijk blijven. Bovendien is er de vereiste dat eenmaal het sjabloon gekend is, de pixelwaarden slechts één maal lineair ingelezen worden en gebufferd indien nodig. Dit laat toe dat de techniek extern gepijplijnd wordt.

Contextvorming voor coderen

Stel dat het beeld $m \times n$ pixels x_{ij} bevat met $i = 1..m$ het kolomnummer en $j = 1..n$ het lijnnummer. Stel verder dat het sjabloon T van orde q gegeven wordt door $T = \{(\mu_t, \nu_t);\ t = 1..q\}$, met als causaliteitscriterium $\nu_t < 0$ of tegelijkertijd $\nu_t = 0$ en $\mu_t < 0$ voor elke $t = 1..q$. Dan wordt de context van een pixel x_{ij} expliciet gedefinieerd als de aaneenrijging

$$y_{ij} = x_{i+\mu_1, j+\nu_1}\, x_{i+\mu_2, j+\nu_2} \cdots x_{i+\mu_q, j+\nu_q}.$$

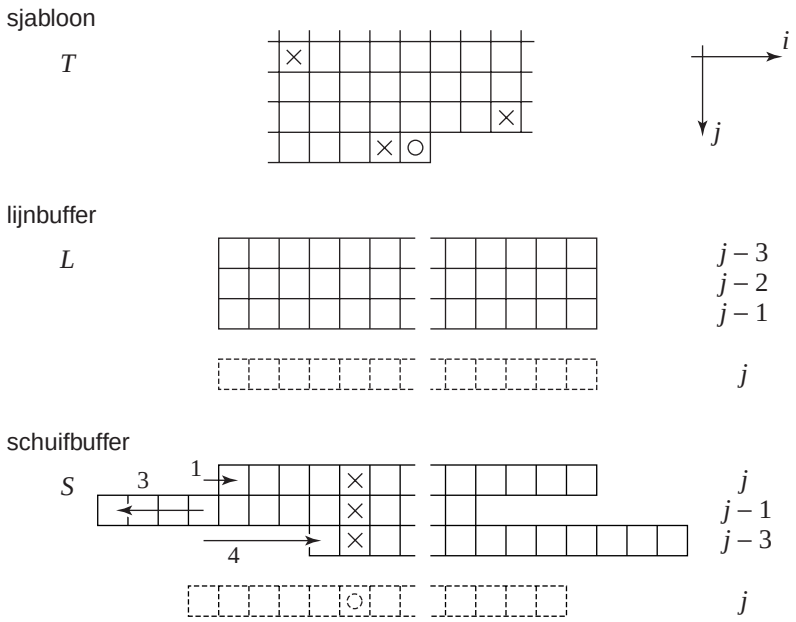
Definiëren we verder μ^- en μ^+ als respectievelijk het minimum en het maximum van alle μ_t, dus $\mu^- = \min\{\mu_t;\ t = 1..q\}$ en $\mu^+ = \max\{\mu_t;\ t = 1..q\}$. Analoog wordt ν^- gedefinieerd als het minimum van alle ν_t, dus $\nu^- = \min\{\nu_t;\ t = 1..q\}$. Er worden geen voorafgaande beperkingen opgelegd aan deze waarden voor de software-implementatie.

Om de geheugentoegang efficiënt te laten verlopen, wordt gebruik gemaakt van twee lijnbuffers. Deze geheugenruimtes bevatten volledige al dan niet verschoven beeldlijnen. De lijnbuffers worden opgezet bij het begin van elke nieuwe lijn en wijzigen niet gedurende het coderen van een lijn.

De eerste buffer, de *lijnbuffer* L, bevat de $-\nu^-$ beeldlijnen boven de huidige lijn j. Die eenvoudige lijnbuffer zorgt ervoor dat elke lijn slechts één keer dient ingelezen te worden en net zolang gebufferd wordt als nodig is.

De tweede buffer, de *schuifbuffer* S, heeft als doel de contextvorming te versnellen door de referentielijnen zorgvuldig te verschuiven overeenkomstig de waarde van het contextsjabloon. Die schuifbuffer haalt de referentielijnen niet rechtstreeks uit de beelddata, maar maakt gebruik van de lijnbuffer L of van de huidige lijn. Voor een gegeven sjabloon T van orde q wordt de schuifbuffer S als volgt opgebouwd. De buffer bevat exact q lijnen, waarbij elke lijn S_t van S overeenkomt met het sjabloonpixel (μ_t, ν_t) met $t = 1..q$. Alvorens lijn j gecodeerd wordt, wordt lijn S_t geïnitialiseerd met de referentielijn $j + \nu_t$, verschoven naar links over een afstand μ_t. Indien $\mu_t < 0$, dan komt dit effectief overeen met een verschuiving naar rechts. Het nettoresultaat is dat alle nodige referentielijnen in de schuifbuffer aanwezig zijn en dat ze op zo'n manier verschoven zijn dat de horizontale verschuiving gecompenseerd is. De context y_{ij} van pixel x_{ij} kan nu eenvoudig gegenereerd worden als de aaneenrijging $S_1(i)\, S_2(i) \cdots S_q(i)$. De volledige schuifbuffer kan geconstrueerd worden alvorens de eerste pixel x_{0j} van een lijn te coderen, zelfs indien sjabloonpixels op de huidige lijn gebruikt worden. Merk op dat indien meerdere sjabloonpixels op dezelfde lijn gelegen zijn, deze lijn meerdere keren zal voorkomen in de schuifbuffer S.

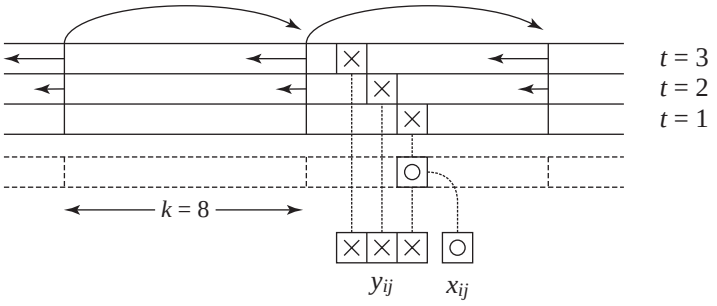
Beschouw het voorbeeld waarbij $T = \{(-4, -3), (3, -1), (-1, 0)\}$. Dan is $q = 3$, $\mu^- = -4$, $\mu^+ = 3$ and $\nu^- = -3$. De lijnbuffer L bevat de $-\nu^- = 3$



Figuur 5.12: Voorbeeld van een sjabloon en de constructie van de twee lijnbuffers. Lijnbuffer L bevat de $-\nu^-$ voorgaande lijnen. Schuifbuffer S bevat de q referentielijnen $j + \nu_t$, verschoven over een afstand μ_t naar links. Het resultaat van deze operatie is dat de context voor de huidige pixel (gemarkeerd als "$\circ$") verticaal kan uitgelezen worden (gemarkeerd als "$\times$").

voorgaande lijnen met als index $j-1$, $j-2$ en $j-3$. De schuifbuffer bevat de volgende $q = 3$ lijnen: lijn $j-3$ verschoven naar rechts over 4 pixels, lijn $j-1$ verschoven naar links over 3 pixels en de huidige lijn j verschoven naar rechts over één pixel. Figuur 5.12 geeft dit sjabloon en de twee buffers grafisch weer. De opeenvolgende contexten y_{ij} voor $i = 1..n$ zijn uit te lezen als kolommen van S.

Om optimaal gebruik te maken van de 32-bit architectuur van de meeste hedendaagse processoren, worden de pixels voorgesteld aan de hand van één bit per pixel en gegroepeerd in blokken van $k = 32$ bits. Het horizontaal verschuiven van een lijn gebeurt op basis van deze blokken. Elk blok wordt opgesplitst in een stuk dat binnen de blokgrenzen blijft en een stuk dat de blokgrenzen overschrijdt. Het eerste stuk wordt verschoven, het tweede stuk wordt verplaatst naar het volgende blok en verschoven over een complementaire afstand. Op deze manier is de tijd nodig om de schuifbuffer op te zetten verwaarloosbaar ten opzichte van de rest van het codeerproces.

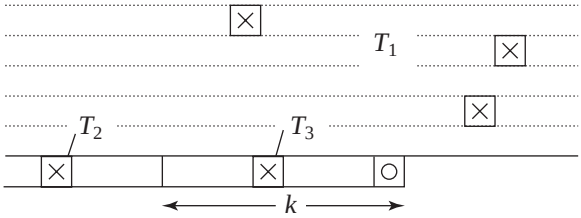


Figuur 5.13: Elk blok van S_t wordt cyclisch verschoven naar links over een afstand $t-1$. In dit voorbeeld is $k = 8$, maar normaal zal $k = 32$ voor normale processoren. De volle pijlen illustreren de cyclische blokverschuiving. De streeplijnen tonen hoe de context opgebouwd wordt door de individuele bits naar beneden te laten zakken.

Tot slot vereist het coderen van een blok het uitlezen van k verticale kolommen van S, elk van hoogte q. Deze operatie is equivalent aan het transponeren van een matrix bestaande uit $q \times k$ bits. Dit is geen voor de hand liggende bewerking voor een moderne processor omdat de instructieset geoptimaliseerd is voor gehele getallen. Experimenteel onderzoek heeft uitgewezen dat dit sneller kan verlopen door een additionele cyclische blokverschuiving toe te passen binnen de blokken van S_t. Elk blok van k pixels dat behoort tot lijn S_t wordt cyclisch verschoven naar links over een afstand $t-1$, en dit voor alle $t = 1..q$. Een cyclische verschuiving betekent dat de pixels links uit het blok geschoven worden er rechts weer binnenkomen. Figuur 5.13 illustreert dit proces voor het sjabloon uit het voorbeeld. Als gevolg van deze additionele verschuiving wordt de context geconstrueerd door de respectieve contextbits naar beneden te laten vallen en in te schuiven in de contextbuffer. Deze "drop-down" operatie voor één contextbit komt neer op één bitsgewijze AND-operatie met een masker dat slechts één '1'-bit telt en een OR-operatie met het voorlopige resultaat. De snelheidswinst door deze cyclische blokverschuiving is te verklaren doordat k individuele bitverschuivingen vervangen worden door één cyclische blokverschuiving van k bits. De snelheidstoename als gevolg van deze additionele cyclische blokverschuiving bedraagt circa 20%.

Contextvorming voor decoderen

Als gevolg van het kritische datapad voor decompressie kan de geoptimaliseerde implementatie voor de contextvorming bij coderen niet zomaar gebruikt worden voor het decoderen. Enkel indien geen sjabloonpixels op de huidige



Figuur 5.14: Afhankelijk van de lengte van het datapad wordt een sjabloon T opgesplitst in drie subsjablonen T_1, T_2 en T_3.

lijn gebruikt worden, dus als $\nu_t < 0$ voor $t = 1..q$, kan bovenstaande aanpak zondermeer toegepast worden. Dit is een onwaarschijnlijke situatie en we gaan dan ook in op de noodzakelijke wijzigingen die moeten aangebracht worden. Omdat de lijnbuffer L enkel voorgaande lijnen bevat die volledig gekend zijn bij het begin van een nieuwe lijn, blijft zijn waarde ongewijzigd.

Afhankelijk van de lengte van de terugkoppeling in het datapad, kunnen de pixels van elk sjabloon T opgedeeld worden in drie subsjablonen:

- Een eerste subsjabloon T_1 bevat de sjabloonpixels die behoren tot de voorgaande lijnen, dus waarvoor $\nu_t < 0$. Deze sjabloonpixels geven aanleiding tot contextpixels die gekend zijn bij het begin van een nieuwe lijn en hebben dan ook het minst kritische datapad.

- Een tweede subsjabloon T_2 bevat de sjabloonpixels die behoren tot de huidige lijn, maar die een bloklengte k of meer verwijderd zijn, dus waarvoor $\nu_t = 0$ en $\mu_t \leq -k$. Deze sjabloonpixels geven aanleiding tot contextpixels die niet gekend zijn voor de gehele lijn, maar enkel voor het volgende blok.

- Een derde subsjabloon T_3 bevat de sjabloonpixels die eveneens behoren tot de huidige lijn, maar dichter dan een bloklengte k gelegen zijn, dus waarvoor $\nu_t = 0$ en $\mu_t > -k$. Deze sjabloonpixels kennen het meest kritische datapad.

Voor het typische geval waar $k = 32$ zal T_2 meestal geen pixels bevatten, maar dit subsjabloon blijft noodzakelijk om een veralgemening mogelijk te maken. Figuur 5.14 illustreert de opsplitsing van een arbitrair sjabloon.

De schuifbuffer S wordt lichtjes anders opgesteld om rekening te houden met de nieuwe subsjablonen. Definieer q_α als het aantal pixels in het subsjabloon T_α. Wegens het kritische datapad van T_3 worden deze sjabloonpixels niet in de schuifbuffer opgenomen. De schuifbuffer bevat slechts $q_1 + q_2$ lijnen

overeenkomstig subsjablonen T_1 en T_3. De verschoven lijnen corresponderend met T_1 zijn volledig gekend alvorens de eerste pixel van een nieuwe lijn gedecodeerd wordt. De verschoven lijnen corresponderend met T_2 zijn leeg bij het begin van een nieuwe lijn maar worden blok per blok ingevuld zoals ze door het decodeerproces gegenereerd worden. Het kritische datapad corresponderend met T_3 kan tot één bit kort zijn, wat een voortdurende aanpassing van de schuifbuffer zou vereisen. Dit is dan ook de reden waarom dit subsjabloon niet aangewend wordt de constructie van S.

Het berekenen van het contextgedeelte voor de sjabloonpixels van T_1 en T_2 is identiek aan het codeerproces. Voor de sjabloonpixels van T_3 wordt een speciaal register van k bits gebruikt dat het *bypassblok* genoemd wordt. Het bevat de k meest recent gedecodeerde pixels en wordt gebruikt voor het berekenen van het contextgedeelte voor de sjabloonpixels van T_3. Elke nieuwe gedecodeerde pixel wordt onmiddellijk in dit bypassblok ingeschoven. Telkens een volledig blok beschikbaar is, wordt het gekopieerd naar de voorlopig gereconstrueerde huidige lijn en naar elk van de lijnen S_t die corresponderen met sjabloonpixels in T_2.

Voor het voorgaande voorbeeld is $T_1 = \{(-4,-3),(3,-1)\}$, is $T_3 = \{(-1,0)\}$ en is T_2 leeg. Het schuifregister bevat twee verschoven lijnen en het bypassblok is leeg bij het begin van elke nieuwe lijn. De contextvorming gebeurt in twee stappen. In een eerste stap genereert dezelfde aanpak als voor het codeerproces de eerste twee contextbits. In een tweede stap wordt de zonet gedecodeerde pixel uit het bypassblok gelezen en toegevoegd als derde contextbit. De context wordt doorgestuurd naar de QM-coder die de pixelwaarde terugstuurt. Deze nieuwe gereconstrueerde pixel wordt onmiddellijk in het bypassblok ingeschoven. Na k pixels wordt het bypassblok gekopieerd naar de voorlopig gereconstrueerde huidige lijn.

5.4.3 Contextvorming — hardware

Het implementeren van de contextvorming in hardware is onderworpen aan dezelfde intrinsieke beperkingen als in software. Het sjabloon is niet gekend tijdens de ontwerpfase en het optimaliseren is dan ook niet triviaal. Twee zaken zijn wezenlijk verschillend voor een ontwerp in hardware: er is de mogelijkheid om specifieke instructies te creëren en de vereiste geheugenruimte voor de buffers moet expliciet begroot worden. Het eerste geeft aanleiding tot instructies die geoptimaliseerd zijn voor het verwerken van bits. Het tweede vereist de specificatie van een bovengrens voor de orde q, de lijnbreedte m en de relatieve verticale positie ν.

Het hardware-ontwerp legt de nadruk op de contextvorming. De Qx-

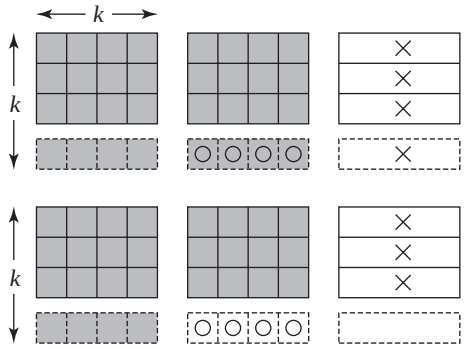
coder, dit is de aritmetische coder met ingebouwd statistisch model, is al commercieel beschikbaar als een ASIC-kern [230]. Deze is gebaseerd op de oorspronkelijke Q-coder die ontworpen was met het oog op een hardware-implementatie [7]. Om deze reden hebben we enkel een gedragsbeschrijving in VHDL geïmplementeerd voor het QM-gedeelte. Omwille van de geheugenvereisten op de ASIC worden vooraf volgende beperkingen opgelegd aan het ontwerp: (1) het maximaal aantal sjabloonpixels is $q = 12$; (2) de relatieve positie (μ_t, ν_t) van elk sjabloonpixel t is beperkt tot $-\delta \leq \mu_t < \delta$ en $-\delta < \nu_t \leq 0$ met $\delta = 32$; en (3) de maximale lijnbreedte m is beperkt tot 32 Kbit. De combinatie van deze beperkingen stelt een compromis voor tussen geheugenvereisten en compressievermogen en het begrenst de buffergrootte tot 128 Kbyte. Het ontwerpproces kan herhaald worden voor andere ontwerpparameters. We beschrijven achtereenvolgens de architectuur en de generatie van de microcode, waarbij het coderen en het decoderen samen aan bod komen.

Architectuur

Het ontwerp voor het contextvormingsgedeelte is gebaseerd op een techniek die *softwarepijplijnen* genoemd wordt. Deze techniek groepeert en herschikt de benodigde individuele bewerkingen voor het genereren van de contexten. De lijnen worden opgesplitst in blokken van lengte k en elke programmacyclus genereert de contexten voor het huidige blok van de huidige lijn en laadt tegelijkertijd de benodigde pixelgegevens in voor de verwerking van het volgende blok van de huidige lijn. De beperkingen op de relatieve posities (μ_t, ν_t) bepalen de bloklengte k, wat resulteert in $k = \delta = 32$. Figuur 5.15 illustreert de bloksgewijze verwerking voor $k = 4$.

Om het programma op te bouwen, definiëren we eerst een aantal primitieve instructies op de buffers en de geheugenregisters. Alvorens de eigenlijke contexten te vormen en het beeld te coderen of te decoderen, wordt het contextsjabloon gecompileerd tot het programma dat in de hardware opgeladen wordt. Elke programmalijn is een vorm van microcode, wat betekent dat het één primitieve instructie van elk type kan bevatten. De primitieve instructies binnen de programmalijn worden tegelijkertijd uitgevoerd. Merk op dat er meerdere programma's mogelijk zijn voor een gegeven sjabloon. Het compileren wordt in software uitgevoerd omdat deze taak niet tijdskritisch is en slechts één maal dient uitgevoerd te worden.

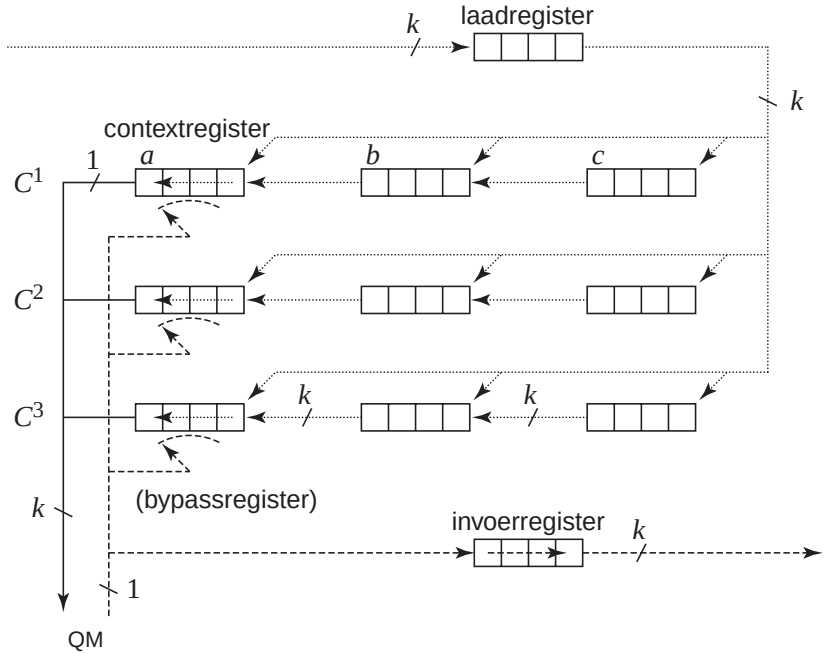
De architectuur bestaat uit q *contextregisters*, een gemeenschappelijk *laadregister* (*E.* loadregister), een gemeenschappelijk *invoerregister* (*E.* inputregister) en een *bypassregister*. Voor elke sjabloonpixel (μ_t, ν_t) met $t = 1..q$ is er precies één contextregister C_t aanwezig, dat op zijn beurt opgesplitst is in



Figuur 5.15: Bloksgewijze verwerking voor coderen (*boven*) en decoderen (*onder*), waarbij de bloklengte $k = 4$ in dit voorbeeld. Eén programmacyclus verwerkt één blok op de huidige lijn (gemarkeerd met "$\circ$"). Referentiepixels gekend aan het begin van elk blok zijn gekleurd in grijs en de blokken die ingelezen worden tijdens de programmacyclus zijn gemarkeerd met "$\times''$.

drie subregisters: een schuifregister C_a^t en twee normale registers C_b^t en C_c^t, kortweg genoteerd als a, b en c. De eerste bits aan de linkerkant van het subregister a vormen samen de huidige context. Subregister b kan zakken in a en subregister c kan op zijn beurt zakken in b. Elk van deze drie subregisters kan onafhankelijk ingeladen worden van het gemeenschappelijke laadregister. Het laadregister wordt op zijn beurt ingevuld met blokken van de huidige lijn of de referentielijnen, dit zijn de blokken gemarkeerd met "$\times$" in figuur 5.15. Het invoerregister buffert de gereconstrueerde pixels gedurende het decoderen. Het bypassregister tenslotte is nodig vanwege het kritische datapad bij decoderen. Het bevat zelf geen gereconstrueerde pixelwaarden maar controleert hoe pas gedecodeerde pixels onmiddellijk terug ingeladen worden in de schuifsubregisters a van de contextregisters. Het bypassregister wordt geïnitialiseerd alvorens het decodeerproces start en omdat het enkel controle-informatie bevat, verandert het niet gedurende het decoderen. Het is noodzakelijk aanwezig voor de reconstructie indien er sjabloonpixels op de huidige lijn gelegen zijn. Figuur 5.16 geeft een concreet voorbeeld van deze architectuur en illustreert de registers voor $q = 3$ en $k = 4$.

Het aantal subregisters waaruit elk contextregister C_t samengesteld is, is een compromis tussen de complexiteit van de compiler, de verwerkingssnelheid en de fysische oppervlakte. De complexiteit wordt bepaald door de gezamenlijke grootte van de normale subregisters die dienen als voorbereidende werkruimte voor de eigenlijke constructie van de contexten, door de randeffecten en door de gewenste optimalisatiegraad. Een complexe compiler is te ver-



Figuur 5.16: In het hart van de architectuur is er een contextregister voor elke sjabloonpixel en een aantal gemeenschapplijke registers. De contexten (volle lijn) worden gevormd uit de referentiepixels (stippellijn). Bij decoderen worden de gereconstrueerde pixels (streeplijn) onmiddellijk opnieuw ingevoerd via het bypassregister.

mijden omdat de correcte werking van het programma moeilijk te verifiëren is voor alle denkbare situaties. Zo zou het enerzijds mogelijk zijn om contextregisters te gebruiken die slechts twee subregisters bevatten: een schuifregister *a* en een normaal register *b*. Maar dit zou het compileren aanzienlijk complexer maken omdat het initiële invullen van de registers moeilijk te optimaliseren wordt. Tegelijkertijd laat het geen hogere verwerkingssnelheid toe. Anderzijds zouden contextregisters die bestaan uit vier of meer subregisters in sommige situaties wel een beperkte versnelling mogelijk maken. De te verwachten winst is evenwel heel gering en daar tegenover staat dat een correcte afhandeling van de randeffecten moeilijker wordt.

Generatie van de microcode

Het vooropstellen van de gewenste architectuur laat toe om de noodzakelijke primitieve instructies te definiëren. Ten eerste is er een instructie “`LOAD`” no-

dig die toelaat om lijnen van de lijnbuffer in te laden in het laadregister, en een instructie "`LOADTO`" om die vervolgens in het contextregister in te laden. Ten tweede is er een instructie "`READY`" die functioneert als een statusvlag en duidelijk maakt dat een context klaarstaat voor uitlezen. Ten derde, indien een subregister a leeg is, is er een instructie "`DROP`" die de subregisters b en c laat zakken in respectievelijk a en b. Tenslotte is er aan het eind van elke programmacyclus tijdens het decoderen een instructie "`STORE`" die de waarde van het invoerregister wegschrijft. Een aantal van deze instructies beschikt over één of meerdere argumenten. Samengevat zijn de volgende primitieve instructies beschikbaar:

- `DROP(`t`)`: verplaats C_b^t naar C_a^t en verplaats C_c^t naar C_b^t;
- `LOAD(`l`)`: kopieer het nieuwe blok van lijn l van de lijnbuffer naar het laadregister;
- `LOADTO(`s`,`t`)`: kopieer het laadregister naar subregister C_s^t;
- `READY`: de huidige context is klaar om naar het QM-gedeelte door te sturen;
- `STORE`: verplaats het invoerregister naar de lijnbuffer.

Deze instructies vormen de bouwstenen voor het programma dat door de compiler gegenereerd wordt op basis van het contextsjabloon. Samengesteld in de gepaste volgorde genereren ze precies k contexten van lengte q in elke programmacyclus.

De initiële invulling van de contextregisters bestaat uit de grijze pixels van figuur 5.15 die links of ter hoogte van elke contextpixel van de eerste pixel in het huidige blok gelegen zijn. Definieer $x \uparrow k$ als $k\lceil (x+1)/k \rceil$, het kleinste veelvoud van k dat strikt groter is dan x en stel verder dat de pixelpositie van de eerste pixel van het huidige blok gegeven wordt door (i, j). Dan wordt voor elke sjabloonpixel (μ_t, ν_t) met $t = 1..q$, het subregister C_a^t ingevuld met de pixels op posities $(i+\mu_t, j+\nu_t)$, $(i+\mu_t+1, j+\nu_t)$, ... tot en met $(i+\mu_t \uparrow k-1, j+\nu_t)$. Vervolgens, indien $\mu_t < 0$ wordt subregister C_b^t ingevuld met de volgende k pixels op positie $(i+\mu_t \uparrow k, j+\nu_t)$ tot en met $(i+\mu_t \uparrow k+k-1, j+\nu_t)$. Het subregister C_c^t is altijd leeg aan het begin van een programmacyclus.

De constructie van het programma wordt gedreven door de volgende twee doelstellingen: (1) lees een context zo snel mogelijk uit van zodra die klaar is, en (2) lees nieuwe blokken eerst in voor de contextregisters die deze het eerst nodig hebben. Op het einde van elke programmacyclus zijn de contextregisters in dezelfde mate ingevuld als aan het begin. Elke cyclus genereert k contexten

en bevat dan ook minstens k programmalijnen, maar meestal iets meer. Het slechtst denkbare sjabloon bevat uitsluitend pixels aan de rechterkant van het toegelaten gebied, dus $T = \{(k-1, -t);\ t = 1..q\}$. In deze situatie moet het programma alle subregisters C_b^t opvullen vooraleer de tweede context kan gegenereerd worden en er zijn dan ook $k + 1 + q$ programmalijnen nodig.

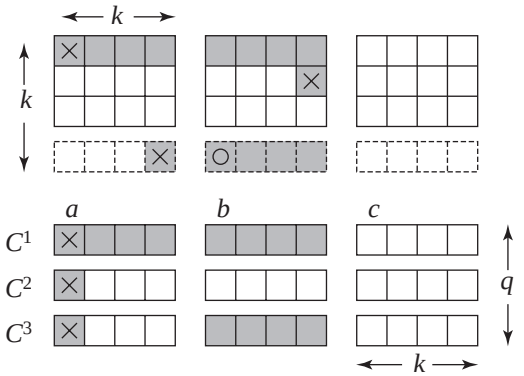
Elke programmalijn bevat ten hoogste één `READY`-, k `DROP`-, één `LOAD`-, één `LOADTO`- en één `STORE`-instructie. Zo kan een `LOADTO` de waarde van het laadregister verplaatsen naar een contextregister terwijl een `LOAD` tegelijkertijd een nieuwe waarde in het laadregister inlaadt. Het samenstellen van deze instructies laat toe om tegelijkertijd een groot aantal bewerkingen in het kader van de vooropgestelde architectuur uit te voeren. De opgegeven beperking $k = 32$ bepaalt het bereik van de argumenten van de instructies en bijgevolg ook de lengte van hun voorstelling. Elke programmalijn of instructiewoord wordt voorgesteld in 29 bits: 1 bit voor de 'end-of-program' instructie, 2 bits voor `READY` (respectievelijk voor coderen en decoderen), 1 bit voor `STORE`, 13 bits voor `DROP`, 6 bits voor `LOAD` en 6 bits voor `LOADTO`.

Het programma wordt softwarematig extern gegenereerd en in de hardware opgeladen als deel van de initialisatie. De door ons ontwikkelde compiler zet een willekeurig sjabloon om in een programma volgens de hierboven beschreven stappen: ten eerste bepaalt het de initiële invulling van de contextregisters en vervolgens genereert het een programma dat zo snel mogelijk de contexten genereert. Het QM-gedeelte is georganiseerd op een gepijplijnde manier wat tot de snelste uitvoering moet leiden.

Voorbeeld

Bij wijze van voorbeeld illustreren we het ontwerp voor hetzelfde voorbeeldsjabloon $T = \{(-4, -3), (3, -1), (-1, 0)\}$ als bij de beschrijving van de software-implementatie. Figuur 5.17 geeft het sjabloon grafisch weer en toont de initiële invulling van de drie contextregisters. Deze invulling stemt overeen met de grijze posities in het sjabloon en de eigenlijke context is uit te lezen als de meest linkse kolom van de subregisters C_a^t.

Figuur 5.18 illustreert de microcode van de programma's voor coderen en decoderen overeenkomstig het voorbeeldsjabloon. Het codeerprogramma telt 5 programmalijnen voor de vorming van $k = 4$ contexten van lengte $q = 3$. De `LOAD`-instructies zijn verantwoordelijk voor het inlezen van de witte rechterblokken uit figuur 5.17. Het decodeerprogramma telt 6 programmalijnen. Het bypassregister zorgt ervoor dat pas gedecodeerde pixels onmiddellijk in de linkerbit van C_a^3 ingeschoven worden. Dit bypassregister wordt eenmalig correct geïnitialiseerd en komt dan ook niet voor in het decodeerprogramma.



Figuur 5.17: Grafische weergave van het contextsjabloon (*boven*) en de hiermee geassocieerde initiële invulling van de contextregisters met $k = 4$ (*onder*). De huidige pixel is gemarkeerd met "$\circ$" en de bijhorende context is gemarkeerd met "$\times$".

De `STORE`-instructie op het einde van de cyclus bewaart het gedecodeerde blok.

5.4.4 Experimentele resultaten

De voorgestelde implementaties voor software en hardware zijn getoetst aan een aantal representatieve testbeelden. De snelheidsresultaten dienen met de nodige omzichtigheid geïnterpreteerd te worden wegens de hoge afhankelijkheid van het testplatform en de testbeelden. Niettemin zijn de resultaten indicatief en tonen ze het potentieel aan van een semi-adaptief sjabloon.

Schatting van de autocorrelatie

De voorgestelde software-implementatie is ontwikkeld op een IBM AIX platform met de bijhorende IBM C-compiler en op twee Linux platforms met de voor de Intel Pentium processor geoptimaliseerde compiler PGCC. De optimalisatie van de compilers wordt ingesteld op het hoogste niveau. Het AIX hardwareplatform heeft een PowerPC 604e processor met een kloksnelheid van 166 MHz, 64 Kbyte L1-cache en 512 Kbyte L2-cache. De Linux hardwareplatforms omvatten een Intel Pentium II Xeon 450 MHz systeem met 512 Kbyte L2-cache en een Intel Pentium II 700 Mhz systeem. Alle experimenteel waargenomen snelheden schalen nagenoeg lineair met de kloksnelheid over het processortype heen. Enige uitzondering hierop is de aard en de grootte van de optimale opzoektabel voor de autocorrelatieschatting, want die is afhankelijk van de grootte van de cache. Alle resultaten die hierna weerge-

Programma: microcode voor coderen

```
1: READY   DROP(3)  LOAD(1)
2:                  LOAD(3)  LOADTO(a,2)
3: READY            LOAD(0)  LOADTO(c,1)
4: READY                     LOADTO(b,3)
5: READY   DROP(1)
```

Programma: microcode voor decoderen

```
1: READY            LOAD(1)
2:                  LOAD(3)  LOADTO(a,2)
3: READY                     LOADTO(c,1)
4: READY
5: READY   DROP(1)
6: STORE
```

Figuur 5.18: Voorbeeld van de uitgewerkte programma's voor coderen en decoderen voor het sjabloon en de contextregisters van figuur 5.17.

geven worden zijn gebaseerd op het Linux Pentium III 700 MHz platform. De andere platforms zijn in het onderzoek opgenomen om processor- en cache-afhankelijke optimalisaties te onderzoeken.

Zoals verwacht schaalt de berekeningstijd voor de schatting van de autocorrelatie lineair met het bereik van i, j, μ en ν. Evaluatie van de autocorrelatieschatting op een typisch gebied van 1024×1024 pixels met $\delta = 32$ stemt overeen met ongeveer 2.2×10^9 bitvergelijkingen. Dit duurt ongeveer 1.95 seconden, wat neerkomt op iets meer dan 10^9 bitvergelijkingen per seconde op het Pentium III 700 Mhz platform. Op het Pentium II Xeon 450 MHz platform duurt dit 2.25 seconden en deze toename in verwerkingssnelheid per klokcyclus is een gevolg van de grotere en snellere $L2$-cache. Ter referentie, een triviale imlementatie die één byte per pixel gebruikt is ongeveer 60–100 keer trager, afhankelijk van het testplatform. De snelle implementatie vereist voldoende geheugen om bij benadering $(\delta + 1)(2\delta + 1)$ beeldlijnen en autocorrelatiestatistieken op te slaan.

Software-implementatie

Voor de implementatie van de contextvorming in software is gebruik gemaakt van dezelfde testplatforms als voor de autocorrelatie. De snelheidsresultaten

Tabel 5.9: Compressiesnelheden voor een aantal representatieve halftoonbeelden op basis van verscheidene sjablonen van variërende orde (in 10^6 bps). De resultaten voor "JBIG-S*" zijn geheel gebaseerd op de implementatie van de JBIG-kit.

sjabloon	ro10k	ro20k	mo10k	mo20k
JBIG-S* 9F-1A*	11.84	12.37	11.85	39.11
JBIG+ 9F-1A (3L)	13.61	14.42	13.31	14.98
JBIG2 12F-1A	12.64	13.31	12.35	13.62
JBIG2 12F-4A	11.70	12.12	11.52	12.62
autocorrelatie 6A	16.05	17.31	13.73	17.73
autocorrelatie 10A	13.53	14.47	12.64	14.78
autocorrelatie 13A	12.47	13.34	11.81	13.59
autocorrelatie 16A	11.57	12.19	11.00	12.37
suboptimaal 6A	16.51	17.43	15.06	18.11
suboptimaal 10A	13.92	14.56	13.36	15.10
suboptimaal 13A	12.84	13.41	12.48	13.89
suboptimaal 16A	11.43	11.64	11.52	12.69

schalen lineair met de kloksnelheid van de processor. De snelheden die hier beschreven worden, beperken zich niet tot de zuivere contextvorming, maar omvatten het gehele codeer- en decodeerproces. De QM-coder wordt overgenomen uit de JBIG-kit, een publiek beschikbare implementatie van JBIG [3]. De waargenomen snelheden zijn afhankelijk van de grootte q van het sjabloon, van de relatieve positie van de individuele sjabloonpixels en van de voorspellende kracht van de contexten.

Tabel 5.9 geeft de codeersnelheid weer voor een aantal testbeelden en sjablonen van tabel 5.2. Alle resultaten behalve die gemarkeerd met "*" zijn behaald met de voorgestelde implementatie. De mate waarin de resultaten afhangen van het beeld, de orde en het sjabloon is beperkt. De meetkundig gemiddelde codeersnelheid over alle beelden en voor alle sjablonen van orde $q = 10$ bedraagt 14.02×10^6 bps. Dit is ongeveer 16% sneller dan de implementatie van de JBIG-kit als het uitzonderlijke beeld "mo20k" niet in rekening gebracht wordt. Het blijkt dat de snelheid van de JBIG-kit heel hoog komt te liggen (tot 40×10^6 bps voor compressie en 50×10^6 bps voor decompressie in onze tests) indien het testbeeld uitzonderlijk goed te comprimeren valt. Het is opmerkelijk dat de voorgestelde implementatie met een volledig vrij sjabloon het voor alle andere beelden sneller doet dan de JBIG-kit waar slechts één

[3] van M. Kuhn (http://www.cl.cam.ac.uk/~mgk25/download/jbigkit-1.2.tar.gz)

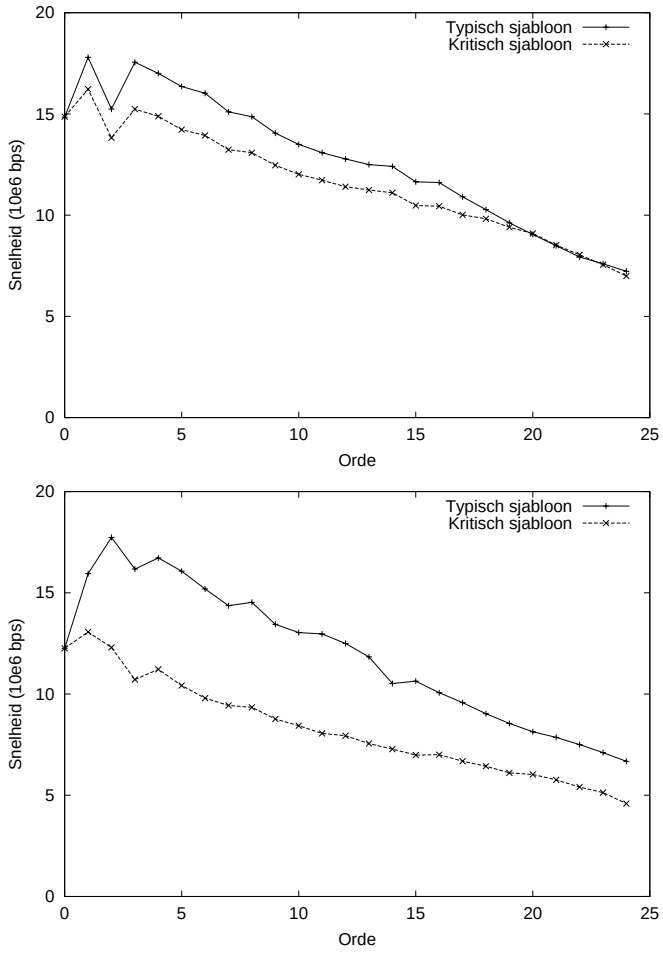
Tabel 5.10: Decompressiesnelheden voor dezelfde omstandigheden als in tabel 5.9. De resultaten voor "JBIG-S*" zijn geheel gebaseerd op de implementatie van de JBIG-kit.

sjabloon	ro10k	ro20k	mo10k	mo20k
JBIG-S* 9F-1A*	15.04	16.60	15.00	49.04
JBIG+ 9F-1A (3L)	11.49	12.26	11.24	12.18
JBIG2 12F-1A	10.10	10.63	9.97	11.44
JBIG2 12F-4A	10.23	10.47	9.87	10.31
autocorrelatie 6A	15.17	14.76	11.30	15.14
autocorrelatie 10A	13.01	12.53	10.80	12.69
autocorrelatie 13A	11.84	11.75	10.28	11.67
autocorrelatie 16A	10.00	10.52	9.43	9.95
suboptimaal 6A	13.97	14.83	12.60	13.85
suboptimaal 10A	12.07	12.62	11.35	12.37
suboptimaal 13A	11.32	11.41	10.06	11.17
suboptimaal 16A	8.94	8.76	9.35	10.21

AT-pixel toegelaten wordt.

Tabel 5.10 geeft de decodeersnelheid weer voor dezelfde testbeelden en sjablonen. Ook hier zijn alle resultaten behalve die gemarkeerd als "*" behaald met de voorgestelde implementatie. De meetkundig gemiddelde snelheid over alle beelden en voor alle sjablonen van orde $q = 10$ bedraagt hier 12.01×10^6 bps, wat bijna 25% lager is dan de implementatie van de JBIG-kit als andermaal het uitzonderlijke beeld "mo20k" niet in rekening gebracht wordt. Een vergelijking tussen beide tabellen leert dat decoderen ongeveer 15% trager verloopt dan coderen, wat te verklaren is door het bypassblok en het kritische datapad. Al kunnen we niet verklaren waarom de JBIG-kit in staat is sneller te decoderen dan te coderen. Tenslotte toont het profileren van de implementatie aan dat het QM-gedeelte ongeveer 40% van de verwerkingstijd vraagt.

Het slechtst denkbare sjabloon voor de voorgestelde implementatie bevat enkel sjabloonpixels op de huidige lijn die direct links van de huidige pixel gelegen zijn, dus $T = \{(-t, 0);\ t = 1\,..\,q\}$. De snelheidsresultaten voor dit kritisch sjabloon en voor het autocorrelatiegebaseerd sjabloon voor het "ro10k" beeld zijn grafisch weergegeven in figuur 5.19. Bij de interpretatie is het belangrijk om rekening te houden met het feit dat de globale snelheid via het QM-gedeelte ten dele afhankelijk is van de compressieverhouding en dat deze negatief beïnvloed wordt door het kritische sjabloon. Bij compressie is dit de enige invloed op de waargenomen snelheid, die dan ook maar weinig lager ligt



Figuur 5.19: Snelheidsresultaten voor "ro10k" (in 10^6 bps) voor het slechtst denkbare sjabloon ("kritisch") en voor het autocorrelatiegebaseerd sjabloon ("typisch"). Als gevolg van het kritisch datapad is het effect van het kritisch sjabloon voor compressie (*boven*) veel minder uitgesproken dan voor decompressie (*onder*).

voor het kritisch sjabloon (circa 10%). Maar bij decompressie manifesteert het kritisch datapad zich en dit heeft een veel grotere impact op de waargenomen snelheid die nu aanzienlijk lager ligt voor het kritisch sjabloon (circa 35%). De figuren tonen ook duidelijk aan hoe de snelheden afnemen bij toenemende orde. Merk op dat het mogelijk is om kritische sjablonen als een uitzondering te implementeren die op het moment van compileren expliciet geoptimaliseerd worden.

Hardware-implementatie

Het ontwerp van de hardware-implementatie is uitgevoerd tot op synthesenivеau met Synopsis. De resultaten van de synthese zijn geverifieerd met een simulator. Deze simulaties hielden geen rekening met de exacte informatie omtrent plaatsing en routering. Als technologie werd gekozen voor SGS Thomson HCMOS7, een 0.25 μ standaard-cel-technologie die tot zes metaallagen ondersteunt en de beoogde klokfrequentie bedraagt 400 Mhz. De totale oppervlakte voor de contextvorming bedraagt slechts 0.68 mm^2. Dit laat meer dan voldoende ruimte over voor de Qx-kern, voor het geheugen voor het statistisch model en voor het nodige I/O-bufferen. De Qx-coder kan gemiddeld 0.85 contexten per klokcyclus verwerken [122, 230]. De snelheden die hierna weergegeven worden hebben betrekking op het gehele ontwerp inclusief contextvorming en Qx-coder.

De waargenomen snelheid hangt af van de aard van het sjabloon. Voor typische sjablonen telt het programma k of net iets meer programmalijnen, met $k = 32$. Dit betekent dat nagenoeg elke instructiecyclus een context genereert. In deze situatie draait de Qx-coder aan zijn volle potentieel, wat toelaat het beeld te coderen aan een snelheid van circa 340×10^6 pixels per seconde. Voor decoderen heeft het kritisch datapad voor contextpixels op de huidige lijn een negatieve invloed. Zo vereist het uiterst nuttige sjabloonpixel $(-1, 0)$ dat het QM-gedeelte gemiddeld één klokcyclus wacht alvorens de pas gedecodeerde pixel ter beschikking staat voor de generatie van een nieuwe context. Als gevolg hiervan halveert de snelheid bij benadering tot 170×10^6 pixels per seconde. Merk op dat q veel kleiner is dan k voor typische sjablonen en bloklengtes en dat de programmalengte bijgevolg min of meer onafhankelijk is van de orde q.

Zoals eerder beschreven wordt één van de slechtst denkbare sjablonen voor coderen gegeven door $\{(k-1, -t);\ t = 1..q\}$. Alle sjabloonpixels behoren tot de meest rechtse kolom van de toegelaten sjabloonruimte en het programma telt $k + 1 + q$ lijnen. Dit meest kritische sjabloon beperkt de codeersnelheid tot 284×10^6 bps voor orde 12. Voor decoderen bestaat het slechtst denkbare sjabloon uit de referentiepixel met het meest kritische datapad $(-1, 0)$, aangevuld met sjabloonpixels uit dezelfde rechterkolom. Dit leidt bijvoorbeeld tot een sjabloon van de vorm $T_c^q = \{(-1, 0), (k-1, -t);\ t = 1..q-1\}$. Dit meest kritische sjabloon leidt tot een decodeersnelheid van 140×10^6 bps voor orde 12. Het valt op te merken dat deze kritische sjablonen een lage voorspellende waarde hebben en daarom weinig relevant zijn.

Tabel 5.11 vat de programmalengtes en de gesimuleerde snelheidsmetingen samen voor twee typische en twee kritische sjablonen. De typische sjablonen omvatten een sjabloon $T_t^6 = \{(-1, 0), (-2, 0), (0, 1), (10, 3), (-3, 11),$

Tabel 5.11: Programmalengtes en experimenteel waargenomen verwerkingssnelheden op basis van simulaties voor de voorgestelde hardware-implementaties (in 10^6 bps).

sjabloon	compressie		decompressie	
	lijnen	snelheid	lijnen	snelheid
I - Orde $q = 6$				
typisch	32	340	33	170
kritisch	39	328	39	164
II - Orde $q = 12$				
typisch	32	340	33	170
kritisch	45	284	45	140

$(-12, 12)\}$ van orde 6 en een sjabloon $T_t^{12} = \{(-1, 0), (-2, 0), (-3, 0), (-1, -1), (10, -2), (11, -3), (10, -3), (-14, -7), (-14, -8), (-3, -11), (-12, -12)\}$ van orde 12. De kritische sjablonen zijn gegeven door T_c^6 en T_c^{12}. De tabel illustreert duidelijk het snelheidsverschil tussen typische en kritische sjablonen enerzijds en tussen compressie en decompressie anderzijds.

De resultaten zijn kwalitatief in lijn maar kwantitatief tot zes keer sneller dan een vergelijkbare JBIG-ABIC implementatie [143]. Een vergelijking van de typische snelheid per klokcylus leert dat de voorgestelde hardware-implementatie relatief gezien ongeveer 45 en 25 keer sneller is dan de voorgestelde software-implementatie voor respectievelijk compressie en decompressie.

De voorgestelde aanpak van een volledig vrij contextsjabloon vertoont sterke overeenkomst met de later voorgestelde *Dispersed Reference Compression* of DRC techniek [208]. Deze is gebaseerd op *ontwikkelende hardware* (*E.* evolvable hardware) en maakt gebruik van een volledig adaptief in plaats van een semi-adaptief sjabloon. Een genetisch algoritme optimaliseert het contextsjabloon op een regelmatige manier [237]. De experimentele resultaten qua compressie lijken een gelijkaardige winst te vertonen ten opzichte van JBIG als onze voorgestelde techniek, maar er treden convergentieproblemen op bij hogere ordes. Hun voorlopige niet-geoptimaliseerde implementatie is 100 tot 200 keer trager dan onze geoptimaliseerde software-implementatie.

5.5 Contextmodel van variabele orde

Voor halftoonbeelden vormt het semi-adaptief maken van het sjabloon een goed compromis tussen compressie en verwerkingssnelheid. Deze hoge snelheid is haalbaar doordat de lengte van de contexten constant is en doordat de waarde van het sjabloon niet verandert gedurende het verwerken van een beeld. Vervalt de snelheidsvereiste als randvoorwaarde, dan vormen contextmodellen van variabele orde en volledig adaptieve sjablonen een interessante optie. Een variabele-orde-model vormt een uitstekende oplossing voor het probleem van contextverdunning. Een volledig adaptief sjabloon laat toe om verregaand rekening te houden met het niet-stationair karakter van het beeld. Hierna stellen we een variabele-orde-model voor dat met succes gecombineerd kan worden met de voorgestelde semi-adaptieve sjablonen. Het is gedeeltelijk geïnspireerd op gelijkaardig werk bij het totstandkomen van de JBIG-standaarden [147] en toont gelijkenissen met andere voorgestelde boommodellen [168, 169].

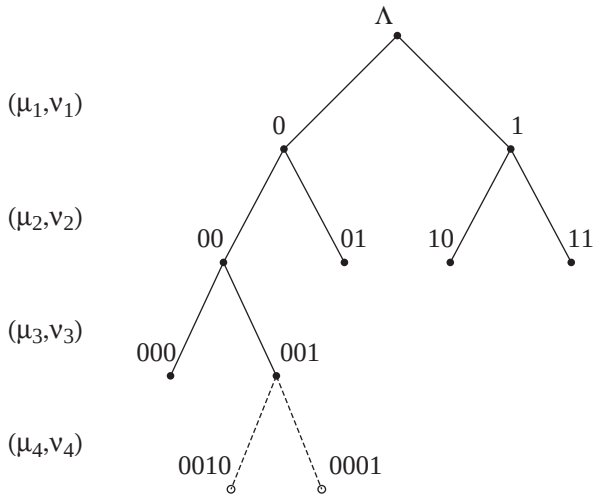
5.5.1 Onvolledige gebalanceerde tweeboom

De contexten van een vaste-orde-model kunnen voorgesteld worden als de bladeren van een *volledige gebalanceerde tweeboom*. Dit is een boomstructuur waarbij elke knoop behalve de eindknopen twee kinderen heeft en waarbij alle eindknopen (of bladeren) dezelfde diepte hebben. Hieruit volgt dat een *onvolledige gebalanceerde tweeboom* (*E.* pruned binary tree) een voor de hand liggend variabele-orde-model wordt. De boom is gebalanceerd omdat iedere knoop ofwel twee ofwel geen kinderen heeft. Daarenboven is de boom onvolledig omdat niet alle eindknopen dezelfde diepte hebben. Het sjabloon is niet langer een constante verzameling van sjabloonpixels, maar wordt nu een geordende rij. Het sjabloon kan ook hier statisch, semi-adaptief of adaptief zijn.

Figuur 5.20 geeft een voorbeeld van een onvolledige gebalanceerde tweeboom. Elke knoop wordt gekenmerkt door een bijhorende context overeenkomstig een onbegrensd maar vast sjabloon $T^{\infty} = (\mu_t, \nu_t)$. Bij elke knoop horen statistieken die de voorwaardelijke distributie $p(x|y)$ beschrijven.

5.5.2 Implementatie

Concreet wordt dit model als volgt geïmplementeerd. Bij het begin van het beeld is enkel de wortel aanwezig die de ledige context Λ beschrijft. Tijdens de verwerking van het beeld groeit de boom en worden statistieken opgebouwd. Omwille van geheugenbeperkingen wordt de boom begrensd door een maximum voor het totaal aantal eindknopen $2^{q_{avg}}$ en door een maximale diepte



Figuur 5.20: Een onvolledige gebalanceerde tweeboom doet dienst als variabele-orde-model. Iedere knoop stemt overeen met een context en bevat dan ook contextafhankelijke statistieken. De streeplijn duidt op het ontstaan van nieuwe kindknopen.

q_{max}. Om het initialiseren van de boom te versnellen wordt gestart met een volledige gebalanceerde boom van een vooropgestelde minimale orde q_{min}. Elke knoop die behoort bij een context y^k van lengte k bevat de statistieken die de voorwaardelijke waarschijnlijkheid $p(x|y^k)$ beschrijven. In een praktische implementatie worden de statistieken niet expliciet bijgehouden, maar gebeurt dit impliciet aan de hand van de gediscretiseerde toestanden van de QM-coder. Indien de knoop geen eindknoop is, bevat deze ook twee wijzers naar de kindknopen behorend bij de contexten y^k0 en y^k1. Een eindknoop wordt *significant* genoemd indien de bijhorende context frequenter is opgetreden dan een voorgedefinieerde drempelwaarde n_s.

Het verwerken van een nieuwe pixel x_{ij} gebeurt in twee stappen. Ten eerste wordt de boom doorkruist overeenkomstig de context y_{ij} totdat een eindknoop bereikt wordt. De waarde van de context volgt uit de pixelwaarden en het sjabloon T^∞. Voor deze eindknoop zijn er twee mogelijkheden: de knoop heeft geen kinderen wegens de boombeperkingen of de knoop heeft geen kinderen omdat deze nog niet significant is. De statistieken horend bij de knoop en de eigenlijke pixelwaarde van x_{ij} worden naar een entropiecoder gestuurd. Ten tweede moet de boom aangepast worden aan de zojuist waargenomen gebeurtenis $y^k x$. Indien de knoop zonet significant geworden is, worden twee kindknopen gegenereerd zolang dit niet conflicteert met de beperkingen van

de boom. Indien geen nieuwe kindknopen ontstaan, worden de statistieken horend bij de eindknoop aangepast. De statistieken van een kindknoop kunnen gelijkmatig geïnitialiseerd worden, of er kan een *erfenismechanisme* (*E.* inheritance mechanism) geïmplementeerd worden waarbij de kindknopen de statistieken van de vaderknoop erven.

Onder typische omstandigheden zal het groeien van de boom beperkt blijven tot het begingedeelte van de boom. Daarna zullen enkel de statistieken van de eindknopen aangepast worden. Bij sterk niet-stationaire beelden kan het aangewezen zijn om de boom te *snoeien* (*E.* prune). Hierbij worden op regelmatige basis alle eindknopen verwijderd tot op het niveau van de vaderknopen. De boom halveert in grootte en groeit dan opnieuw aan tot volle grootte. Bij de implementatie van dit model hebben we geen aandacht besteed aan de snelheid van de implementatie en die is dan ook enkele grootte-ordes trager dan het vaste-orde-model.

5.5.3 Experimentele resultaten

Tabel 5.12 geeft de resultaten weer voor zowel het vaste-orde-model als het variabele-orde-model, waarbij elke techniek exact 1024 eindknopen kent om een faire vergelijking toe te laten. Rekening houdend met de grootte van de beelden zijn de optimale parameters qua maximale en gemiddelde diepte van de boom aanzienlijk hoger. De compressieresultaten in de tabel zijn dan ook een stuk lager dan wat haalbaar is indien de verwerkingstijd en de hoeveelheid werkgeheugen ongelimiteerd zouden zijn. De resultaten voor de onvolledige gebalanceerde tweeboom zijn gebaseerd op geoptimaliseerde parameters.

De sjablonen voor de technieken die gebruik maken van een onvolledige gebalanceerde tweeboom zijn opgebouwd uit een onbegrensde lijst contextpixels. Concreet betekent dit het volgende voor de respectieve technieken. Het sjabloon gebaseerd op de L_1-afstand is beeldonafhankelijk en sorteert de contextpixels volgens oplopende L_1-afstand. Het sjabloon volgens de optimale JBIG-implementatie gebruikt het beste AT-pixel als eerste contextpixel en vult het sjabloon verder aan met beeldonafhankelijke L_1-pixels. Het sjabloon gebaseerd op autocorrelatie bestaat exclusief uit contextpixels gesorteerd volgens afnemende autocorrelatie waarbij $\delta = 20$ is gesteld. Het suboptimale sjabloon bevat eerst de sjabloonpixels die voortvloeien uit de gretige zoektocht op een stuk van 2048×2048 pixels waarbij $\delta = 24$ en dit wordt verder aangevuld met pixels volgens de L_1-afstand.

De tabel toont duidelijk aan dat het model van variabele orde het consequent beter doet dan een model van vaste orde. De winst varieert grofweg van 2% tot 25%. De winst is groter indien het sjabloon weinig geoptimaliseerd

Tabel 5.12: Compressieverhouding van de referentiehalftoonbeelden aan de hand van een contextmodel van vaste orde (I) en variabele orde (II). Alle technieken gebruiken precies 2^{10} eindknopen om een faire vergelijking toe te laten. De notatie "$\approx x$F-yA+Z" staat voor gemiddeld $x + y$ eindknopen waarvan x gebaseerd op vaste contextpixels en y op adaptieve, verder aangevuld met vaste (Z=F) of adaptieve (Z=A) adaptieve pixels. De resultaten voor de onvolledige gebalanceerde tweebomen zijn gebaseerd op geoptimaliseerde parameters.

techniek	ro5k	ro10k	ro20k	mo10k	mo20k
I - Model van vaste orde					
L_1-afstand 10F	5.57	4.10	6.50	4.19	13.42
JBIG+ 9F-1A (3L)	7.14	5.59	9.34	4.40	15.15
autocorrelatie 10A	8.47	6.78	11.09	3.29	12.43
suboptimaal 10A	9.12	7.08	11.85	4.30	19.97
II - Onvolledige gebalanceerde tweeboom					
L_1-afstand ≈10F+F	5.98	4.70	8.19	5.24	16.27
JBIG+ ≈1A-9F+F	7.47	5.98	10.33	5.24	16.27
autocorrelatie ≈10A+A	8.71	6.80	11.55	3.81	12.95
suboptimaal ≈10A+AF	9.47	7.23	12.36	5.47	21.86

is, wat zich vooral manifesteert voor het sjabloon gebaseerd op de L_1-afstand en het geoptimaliseerde JBIG-sjabloon. Voor het sjabloon gebaseerd op de autocorrelatie is de winst kleiner omdat de boom vooral groeit in de richting van de uiterste takken waar alle pixels ofwel waarde '0' ofwel waarde '1' hebben. Het suboptimale sjabloon heeft als nadeel dat het geoptimaliseerd is voor een vaste-orde-model en dat de uitbreiding gebruik maakt van niet-optimale contextpixels gebaseerd op de L_1-afstand.

De winst is ook groter voor de stochastische dan voor de klassieke halftoonbeelden. De oorzaak ligt bij het feit dat er een complexere microstructuur aanwezig is in deze beelden en dit vereist een grotere context. Dit bleek ook al uit de grafieken van figuur 5.8 waarop duidelijk te zien is hoe de compressieverhouding gestaag blijft stijgen bij toenemende orde.

Tabel 5.13 geeft de optimale parameters weer voor de compressieresultaten van voorgaande tabel. De kandidaatwaarden voor de maximale orde zijn $q_{max} \in \{10, 12, 14, 16, 18, 20\}$, voor de minimale orde zijn die $q_{min} \in \{10, 8, 6, 4, 2, 0\}$. Verder beperkt de drempelwaarde zich tot $n_s \in \{64, 128, 256, 512, 1024\}$ en kan het erfenismechanisme $i \in \{0, 1\}$ al of niet geactiveerd worden. Experimenten voorafgaand aan de eigenlijke parameter-

Tabel 5.13: Optimale parameterwaarden voor de onvolledige tweeboom voor een aantal halftoonbeelden. De waarden in de notatie "q_{max}-q_{min}-n_s-i" stellen respectievelijk de maximale orde q_{max}, de minimale orde q_{min}, de drempelwaarde n_s voor significantie en het erfenismechanisme i voor. De gemiddelde orde is vooraf gelijkgesteld aan $q_{avg} = 10$.

techniek	ro5k	ro10k	ro20k	mo10k	mo20k
L_1	20-4-128-0	20-6-64-1	20-4-256-0	16-6-1024-1	14-6-256-1
JBIG+	16-6-128-0	20-6-64-1	20-6-128-1	16-6-1024-1	14-6-256-1
autoc.	14-6-256-0	14-6-1024-1	14-2-128-1	20-8-256-1	12-2-256-1
subopt.	14-6-256-1	12-6-1024-0	14-6-256-1	18-8-128-1	14-4-1024-1

optimalisatie tonen aan dat het regelmatig snoeien van de boom een negatief effect heeft op het uiteindelijke resultaat. Dit is een gevolg van de grote mate van adaptiviteit die al in de QM-coder ingebakken is maar door het snoeien deels teniet gedaan wordt.

Al is voorzichtigheid geboden bij het interpreteren van de resultaten, toch leert de tabel het volgende. Het invoeren van een minimale orde en een maximale is een zinvolle beperking voor de binaire boom. De adaptiviteit ingebakken in de QM-coder heeft op zich een positief effect, dat ten dele verloren gaat indien de contextboom te sterk verfijnt. Het duurt dan gemiddeld een stuk langer alvorens dezelfde context terug optreedt en de bijhorende statistieken laten zich veel moeilijker aanpassen aan de niet-stationaire gegevens. Vooral voor de klassieke halftoonbeelden met de twee volledig semi-adaptieve sjablonen is de externe groeibeperking vrij strikt, wat aanleiding geeft tot een boom met een meer evenwichtige diepte. Voor de andere sjablonen heeft de binaire boom er baat bij zo snel mogelijk behoorlijk diep uit te groeien naar de meest voorkomende contexten. De resultaten voor de stochastische gerasterde beelden zijn andermaal moeilijk te interpreteren. Dit gaat vermoedelijk terug op de sterk resolutieafhankelijke microstructureren en de kleine afwijkingen erop. Het erfenismechanisme lijkt een positief effect te hebben, maar de resultaten zijn niet altijd even duidelijk.

De variabele diepte van de context heeft een negatief effect op de contextgeneratiesnelheid. De context wordt bit per bit opgebouwd terwijl telkens een andere tak van de boom verkozen wordt. Voor een standaardimplementatie is de verwerkingssnelheid nagenoeg een derde lager dan die van het vaste-orde-model. Maar die snelheid is op zijn beurt nagenoeg twaalf keer lager dan de snelheid van de geoptimaliseerde implementatie. Het optimaliseren van de implementatie van het variabele-orde-model is minder voor de hand liggend

dan voor het vaste-orde-model. Maar zelfs na uitgebreide optimalisatie zal de uiteindelijke snelheid beduidend lager zijn.

5.6 Uitbreiding naar niet-binaire residu's

Het adaptief maken van het contextsjabloon opent de weg naar een nieuwe familie van universele compressieschema's. Omdat de binaire QM-coder werkt op het meest atomaire datatype, namelijk de bits van de binaire voorstelling, kan deze gebruikt worden om alle datatypes te coderen. De regelmatige structuur die het natuurlijke datatype in het binaire patroon induceert, wordt geprojecteerd op een datatype-afhankelijk contextsjabloon. Dit laatste reflecteert eigenschappen zoals de woordlengte van het natuurlijk datatype en de dimensionaliteit van de structuur. De kwantitatieve aard van het datatype kan helaas moeilijk door een semi-adaptief contextsjabloon onderschept worden. Daarom is vaak een datatype-afhankelijke voorbewerking nodig. De contextmodellering die daarop volgt is wel universeel. De kracht van deze aanpak ligt niet zozeer in de snelheid of de efficiëntie van de compressie, maar wel in de universaliteit van de globale aanpak en de entropiecoder. De voorgestelde techniek is een gevorderde voortzetting van een gelijkaardige aanpak waarbij contextmodellering rechtstreeks op de al dan niet graygecodeerde bitvlakken wordt toegepast [183]. Het beeldtype induceert enkel wijzigingen in de voorspelling en het contextsjabloon.

5.6.1 Voorspelling en graycodering als voorbewerking

Grijswaarden- en kleurenbeelden vormen een mooi voorbeeld om deze aanpak te illustreren. De voorbewerking decorreleert de beeldwaarden aan de hand van voorspelling. Om de voorspelling eenvoudig te houden, wordt gekozen voor de voorspeller u_7 van LJPEG, die beschreven staat in tabel 4.4. Deze voorspeller beeldt de dichtste buren $x_{i-1,j}$ en $x_{i,j-1}$ af op de voorspelling $\hat{x}_{ij} = (x_{i-1,j} + x_{i,j-1}) \div 2$. Stelt l de woordlengte voor van het datatype, dan doet niet het eigenlijke verschil $x_{ij} - \hat{x}_{ij}$ maar het gecorrigeerde verschil $d_{ij} = (x_{ij} - \hat{x}_{ij} + 2^{l-1}) \bmod l^k$ dienst als verschilbeeld of *residu*. Merk op dat de modulo-bewerking geen obstakel vormt voor de verliesloze reconstructie. Voor kleurenbeelden kan IEP een bijkomende tonale decorrelatie opleveren.

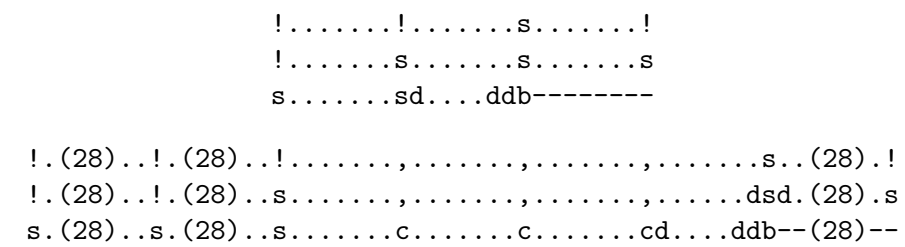
Om de efficiëntie van de binaire contextmodellering verder te verhogen, wordt de graycodering toegepast op de residuwaarde d_{ij}. Dit resulteert in het graygecodeerde residu $g_{ij} = G(d_{ij}) = d_{ij} \oplus (d_{ij} \gg 1)$. De kracht van de graycodering schuilt in de eigenschap dat de binaire graygecodeerde voorstellingen $G(a)$ en $G(b)$ slechts in één bit verschillen indien $|a - b| = 1$.

5.6.2 Sjabloonconstructie

De spatiale en tonale voorspelling slagen er niet in alle redundantie uit de beelden te halen. Binaire contextmodellering vormt hiervoor een aangewezen methode, maar dit moet met de nodige voorzorg gebeuren. Enerzijds bouwt elk van de bitvlakken zelf onafhankelijke statistieken op, maar anderzijds moeten deze statistieken geconditioneerd zijn op de waarnemingen in andere bitvlakken. Dit wordt bereikt door de context y_{ijk} van een residu-bit x_{ijk} op positie (i, j) in bitvlak k samen te stellen uit twee deelcontexten $y^{(1)}$ en $y^{(2)}$. De eerste deelcontext $y^{(1)}_{ijk}$ is niks anders dan de binaire voorstelling van de index k van het bitvlak. Op zijn beurt kan deze samengesteld zijn uit een bitvlakindex en een kleurindex. Deze deelcontext zorgt ervoor dat er één contextmodel per bitvlak en per kleur opgebouwd wordt. De tweede deelcontext $y^{(2)}_{ijk}$ is gebaseerd op een contextsjabloon T, dat zich spatiaal over meerdere dimensies, binair over meerdere bitvlakken en tonaal over meerdere kleuren uitstrekt. Deze deelcontext zorgt ervoor dat elk bitvlakcontextmodel verder opgesplitst wordt volgens de conditionerende bitwaarden. Samengesteld bereikt dit contextmodel de vooropgestelde statistische doelstellingen.

Figuur 5.21 geeft twee voorbeelden van een experimenteel contextsjabloon voor de tweede deelcontext $y^{(2)}$, respectievelijk voor monochrome beelden en voor CMYK-beelden, waarbij de woordlengte $l = 8$ vooropgesteld wordt. De code voor de notatie van dit sjabloon is als volgt: "`b`" staat voor de huidige bit, "`s`" staat voor een contextbit met dezelfde bitvlakindex van dezelfde kleur, "`c`" staat voor een contextbit met dezelfde bitvlakindex maar van een ander kleur, "`d`" staat voor een contextbit in een ander bitvlak, "`.`" staat voor een ongebruikte bit in een ander bitvlak, "`,`" staat voor een ongebruikte bit met dezelfde bitvlakindex maar van een ander kleur, "`!`" staat voor een ongebruikte bit met dezelfde bitvlakindex maar van dezelfde kleur, en "`-`" tenslotte staat voor een niet-causale bit. De notatie "`(x)`" wijst op het weglaten van x contextposities (in casu $x = 28$) uit de voorstelling van het sjabloon. De aanwezigheid van "`s`"-contextbits laat toe de spatiale redundantie uit te buiten die niet door de eenvoudige voorspeller onderschept wordt. De aanwezigheid van "`d`"-contextbits laat toe de redundantie tussen de bitvlakken uit te buiten en de aanwezigheid van "`c`"-contextbits is er op gericht om de overblijvende tonale redundantie uit te buiten.

Samengesteld geven deze contextsjablonen aanleiding tot een sjabloon van respectievelijk orde $q = 12$ (respectievelijk 3 contextbits voor de index en 9 voor het sjabloon) en $q = 19$ (respectievelijk 3 contextbits voor de bitvlakindex, 2 voor de kleurindex en 14 voor het sjabloon). Deze sjablonen zijn experimenteel samengesteld en in zekere mate empirisch geoptimaliseerd. De uiteindelijke efficiëntie verandert evenwel weinig voor gelijkaardige sjablonen,



Figuur 5.21: Experimenteel samengesteld contextsjabloon voor monochrome beelden (*boven*) en voor CMYK-beelden (*onder*), beiden voor $l = 8$ bits per pixel per kleur. De notatie voor deze sjablonen wordt in de tekst uitgelegd.

zolang het kwalitatief aandeel van elk van de factoren maar aanwezig blijft.

5.6.3 Experimentele resultaten

Tabel 5.14 vat enkele experimentele resultaten samen voor twee monochrome testbeelden en één CMYK-testbeeld. Resultaten voor andere beelden vertonen een gelijkaardig gedrag en worden hier niet opgenomen. Ter vergelijking zijn een aantal resultaten met standaardtechnieken uit het vorig hoofdstuk overgenomen. De techniek "FO-RC" staat voor residucodering met een vaste-orde-model van orde $q = 12$ (9 spatiale contextbits en 3 tonale) in combinatie met één van de hiervoor beschreven contextsjablonen. De techniek "BT-RC" staat voor residucodering met een variabele-orde-model van maximale orde $q_{max} = 20$, gemiddelde orde $q_{avg} = 12$ en minimale orde $q_{min} = 6$. Verder is de drempelwaarde voor significantie gegeven door $n_s = 128$ en wordt gebruik gemaakt van het erfenismechanisme. De sjablonen worden verder uitgebreid met afwisselend spatiale en tonale contextbits. De voorspelling is dezelfde als voor LJPEG, voor alle technieken wordt graycodering toegepast op het residu en voor "IEP + FO-RC" wordt de voorspelling aangevuld met IEP.

Uit de tabel volgt dat FO-RC het ongeveer 6–10% beter doet dan LJPEG of JBIG-S voor dezelfde voorspeller en zonder tonale voorspelling. De additionele winst van het variabele-orde-model ten opzichte van een vaste-orde-model is heel beperkt. Dit wijst er op dat er weinig of geen contextverdunning optreedt. Zonder tonale voorspelling is de efficiëntie een stuk lager dan die van JPEG-LS of CALIC, maar de voorspeller is dan ook veel eenvoudiger. Daar tegenover staat dat de voorgestelde techniek veel universeler is, omdat ze kan aangewend worden voor monochrome, driedimensionale, kleuren- en meercomponentenbeelden. Voor kleurenbeelden levert de expliciete tonale voorspelling aan de hand van IEP een aanzienlijke winst op van ongeveer 10%.

Tabel 5.14: Compressieresultaten van universele residucodering op een beperkt aantal testbeelden, uitgedrukt als bitdiepte per pixel per component (bpp). Ter vergelijking worden de resultaten van een aantal standaardtechnieken herhaald.

techniek	lena-l	musicians-c	musicians-cmyk
I - Standaardtechnieken			
LJPEG	4.70	4.87	4.83
JBIG-S	4.72	4.91	4.81
JPEG-LS	4.24	4.42	4.31
CALIC	4.11	4.26	4.15
II - Universele residucodering			
FO-RC	4.43	4.52	4.41
BT-RC	4.41	4.49	4.37
IEP + FO-RC	–	–	3.96
IEP + BT-RC	–	–	3.92

Het resultaat is beter dan voor CALIC, maar de vergelijking is niet gerechtigd omdat CALIC geen gebruik maakt van tonale decorrelatie.

5.7 Samenvatting en eigen bijdragen

Halftoonbeelden komen voor in twee categorieën: klassieke en stochastische. Klassieke halftoonbeelden zijn binaire beelden met de uitzonderlijke eigenschap dat de autocorrelatie niet monotoon afneemt. Als gevolg hiervan is het bij contextmodellering aangewezen gebruik te maken van een semi-adaptief contextsjabloon dat de periodiciteit van de beelden in rekening brengt. Ook voor stochastische halftoonbeelden kan het volledig vrij maken van het sjabloon een positief effect hebben op de efficiëntie. De constructie van een optimaal sjabloon is een heel intensieve operatie. Een gretig zoekproces levert een nagenoeg optimaal sjabloon op maar dit zoekproces duurt te lang voor praktische doeleinden. Het schatten van de autocorrelatie op een representatief stuk van een beeld vormt een snel en goed presterend alternatief.

Voor klassieke halftoonbeelden geeft dit autocorrelatiegebaseerd sjabloon aanleiding tot een winst van ongeveer 50% ten opzichte van JBIG1 in de standaardimplementatie en ongeveer 20% ten opzichte van JBIG1 in de uitgebreide implementatie. Dit is bijna even goed als het moeilijk te bepalen suboptimale sjabloon dat het nog 4–8% beter doet. Voor stochastische halftoonbeelden doet

het autocorrelatiegebaseerd sjabloon het slechter, maar levert het suboptimale sjabloon wel een winst op die kan oplopen tot 40%. Voor beide categorieën ligt de optimale orde aanzienlijk hoger dan 10. De invloed op de efficiëntie van het belichten en opnieuw inscannen van een halftoonbeeld is beperkt. Voor klassieke halftoonbeelden zijn de resultaten vrijwel onafhankelijk van de halftoonparameters, behalve dan van de resolutie.

Van een niet geoptimaliseerde implementatie wordt verwacht dat het semi-adaptief maken van het contextsjabloon een vertraging van het codeerproces tot een factor 4 met zich zal meebrengen. Om deze redenen beperkt JBIG2 het aantal adaptieve pixels in het sjabloon van orde 16 tot 4. Maar voor de geoptimaliseerde implementatie in software is de compressiesnelheid voor het semi-adaptief sjabloon niet lager dan voor een statisch sjabloon. Als gevolg van het kritisch datapad is de decompressiesnelheid voor het semi-adaptief sjabloon wel circa 25% trager dan voor het statisch sjabloon. Op een standaard Pentium III platform (700 Mhz kloksnelheid) bedraagt de compressie- en decompressiesnelheid respectievelijk ongeveer 14×10^6 en 12×10^6 pixels per seconde. Dit toont aan dat de beperking in JBIG2 strikt genomen niet nodig is. Een geoptimaliseerde implementatie in hardware is gebaseerd op softwarepijplijnen. Het semi-adaptief contextsjabloon wordt gecompileerd tot een programma in microcode. De compressie- en decompressiesnelheden bedragen respectievelijk ongeveer 340×10^6 en 170×10^6 pixels per seconde. Een geoptimaliseerde implementatie voor het schatten van de autocorrelatie laat ongeveer 10^9 bitvergelijkingen per seconde toe, wat voor typische parameters aanleiding geeft tot 1–4 seconden voor de generatie van het autocorrelatiegebaseerd sjabloon. Hierbij dienen we op te merken dat er na de officiële publicatie van de JBIG2-standaard binnen het standaardisatiecomité aan gewerkt wordt om de beperking van 4 AT-pixels op te heffen.

Een onvolledige gebalanceerde tweeboom laat toe de beelden efficiënter te modelleren. De compressie verloopt aanzienlijk trager en is bovendien moeilijker te optimaliseren, maar de efficiëntie neemt toe met 2–25%. Het semi-adaptief maken van het sjabloon laat bovendien toe om binaire contextmodellering toe te passen op niet-binaire residu's. Voor monochrome beelden bedraagt de winst voor dezelfde voorspeller als JPEG ongeveer 5–10%. De methode is minder efficiënt dan de meest geavanceerde methode CALIC, maar maakt dan ook gebruik van een voor de hand liggende voorspeller. De grote kracht van de methode schuilt in het feit dat ze universeel toe te passen is op alle beeldcategorieën.

Het gedeeltelijk adaptief maken van het sjabloon gebaseerd op coïncidentie was al ondersteund in de JBIG-standaarden. Het volledig vrij maken van alle sjabloonpixels zonder dat dit de snelheid wezenlijk beïnvloedt is een eigen

bijdrage die geresulteerd heeft in een aantal bijdragen op internationale conferenties [50, 51, 58, 59]. Zowel de autocorrelatiegebaseerde sjabloonconstructie als de geoptimaliseerde software- en hardware-implementatie hebben aanleiding gegeven tot een publicatie in een internationaal tijdschrift [57, 64]. Het onderzoek naar de hardware-implementatie gebeurde in nauwe samenwerking met Dimitri Van De Ville en Frederik Habils. De modelvorming volgens een onvolledige gebalanceerde tweeboom evenals de uitbreiding naar niet-binaire residu's hebben eveneens geresulteerd in een aantal bijdragen op internationale conferenties [47, 50, 62].

Hoofdstuk 6

Contextmodel voor de voorspelling van de value-at-risk

6.1 Inleiding

Uit de voorgaande hoofdstukken is duidelijk gebleken dat contextmodellering in al zijn vormen en gedaanten aanleiding geeft tot zowat de meest efficiënte modellen voor compressie. Bovendien creëert het een heel flexibel en generisch kader dat toelaat rekening te houden met andere aspecten van de gegevens, zoals eventuele niet-stationariteiten, de periodiciteit, het dynamisch bereik en de dimensionaliteit van de te comprimeren gegevens.

Maar alle contextmodellen voor verliesloze compressie worden beperkt door de strikte voorwaarde van reversibiliteit. Enkel de waarnemingen uit het verleden die bovendien deel uitmaken van de te comprimeren gegevens komen in aanmerking voor het schatten van het voorwaardelijk waarschijnlijkheidsmodel. Bovendien komen enkel bewerkingen in aanmerking die exact binair reproduceerbaar zijn op een groot aantal machineplatformen.

Contextmodellen komen ook in aanmerking voor andere toepassingen dan compressie. Overal waar een probabiliteitsdichtheidsfunctie (of pdf) van de bestudeerde gegevens nodig is en waar een eenvormig model tekort schiet, kan contextmodellering een uitkomst bieden. Dikwijls vervalt hierbij de strikte voorwaarde van reversibiliteit wat een nog veel ruimer kader biedt waarbinnen gewerkt kan worden.

In dit hoofdstuk beschrijven we een contextmodel dat geschikt is voor de voorspelling van de value-at-risk van financiële instrumenten. Deze grootheid

is een eenvoudige numerieke statistische maat voor het mogelijk financieel verlies van een gegeven portefeuille onder normale marktomstandigheden. De huidige modellen van de rendementen hebben twee grote tekortkomingen. Ten eerste slagen de gaussiaanse modellen er niet in de nochtans duidelijk aanwezige dikke staarten van de distributies in rekening te brengen. Ten tweede beperken afwijkingen in de stationariteit aanzienlijk de efficiëntie van de value-at-risk-voorspellingen die volgen uit het model. Deze beperkingen komen later meer gedetailleerd aan bod. Er zijn duidelijk parallellen tussen contextmodellering voor compressie en value-at-risk-voorspelling, maar daarnaast steken ook een aanzienlijk aantal tegenstellingen de kop op.

De aanpak van dit hoofdstuk geeft aanleiding tot wat soms *blinde modellering* (*E.* blind modeling) genoemd wordt. De kennis van de financiële wereld die aan het contextmodel toegevoerd wordt, beperkt zich tot historische waarnemingen. Andere kennis zoals de werking van bepaalde goed begrepen marktmechanismen worden niet expliciet in het model ingebouwd. Deze aanpak laat toe om op basis van de resultaten van het contextmodel uitspraken te maken over de verborgen werking van bepaalde marktevoluties. Het is dus tegelijkertijd een werkmiddel om voorspellingen te maken als een methode om de markten te analyseren.

Dit hoofdstuk is als volgt samengesteld. Paragraaf 6.2 beschrijft de achtergrond van risicobeheer in financiële markten en de definitie van de value-at-risk als eenvoudige numerieke statistiek. Vervolgens behandelt paragraaf 6.3 het eigenlijke contextmodel evenals de overeenkomsten en tegenstellingen met de contextmodellen voor compressie. Daarnaast komt een geavanceerde partitionering van de hogerdimensionale contextruimte aan de hand van boomgestructureerde vectorkwantisatie aan bod. In paragraaf 6.4 worden een aantal statistische technieken aangehaald om de efficiëntie van de verschillende modellen en de bijhorende voorspellingen te evalueren. Paragraaf 6.5 bespreekt een aantal experimentele resultaten voor de US S&P500-index. Tenslotte eindigt paragraaf 6.6 met een samenvatting en geeft het de eigen bijdragen weer.

Het onderzoek voorgesteld in dit onderzoek gebeurde onder begeleiding van de vakgroep Financiële Economie en in samenwerking met collega Steven Van Assche. Hij nam het programmeerwerk en de uitvoering van de experimenten op zich, terwijl ondergetekende verantwoordelijk is voor het initiële concept, de literatuurstudie, de publicaties en de wetenschappelijke methodologie. De gedetailleerde uitwerking en de verwerking van de resultaten verliep in onderlinge samenwerking. Het spreekt voor zich dat de individuele bijdrage van elke partij evenals de gezamenlijke discussies onmisbaar waren om tot het uiteindelijke resultaat te komen. De resultaten van dit onderzoek zijn eveneens gepubliceerd in het proefschrift van Steven Van Assche [248].

6.2 De value-at-risk

Om niet tenonder te gaan aan de volatiliteit van de financiële markten investeren zowel banken als niet-financiële bedrijven veel middelen in risicobeheersystemen. Een efficiënter risicobeheersysteem betekent een competitief voordeel en als gevolg hiervan wordt risicobeheer meer en meer een kwantitatieve discipline. Volgens internationale standaarden opgesteld door multinationale organisaties — in het bijzonder de "Bank for International Settlements" of BIS — moeten banken en andere financiële tussenpersonen voldoende kapitaal in huis houden om een aantal potentiële risico's te overleven. De belangrijkste van deze risico's zijn het risico van de tegenpartij, het marktrisico en het interestrisico.

De meeste landen en financiële toezichthouders hebben deze richtlijnen vertaald in hun financiële wetgeving en hun interne voorschriften. De overheid interfereert in de financiële risisocontrole om een voldoende mate van stabiliteit te kunnen garanderen over de financiële instellingen heen. Het grootste risico schuilt in de interne besmettingseffecten van financiële noodsituaties en in het negatieve doorstroomeffect hiervan naar de maatschappij.

De aanpak voor het berekenen van minimale kapitaalbeschikbaarheidsstandaarden voor een stabiele situatie was traditioneel gebaseerd op *regels* (*E*. rules). Dit betekent dat de verschillende risicotypes binnen elke instelling geïdentificeerd en gekwantificeerd worden volgens erkende berekeningsmethoden en dat een vooraf vastgelegde hoeveelheid kapitaal gereserveerd wordt. Het blijkt echter dat deze aanpak meer en meer leidt tot het opstellen van innovatieve financiële contracten waardoor bepaalde risico's migreren naar de risicocategorie die de laagste kapitaalvereisten kent, wat op zijn beurt wetgevende arbitrage vereist [9–11].

Bovendien worden regelgevers en toezichthouders geconfronteerd met een snel wijzigende competitieve financiële omgeving. Hierbij wijzigt zowel de organisatie van de financiële tussenpersonen (bijvoorbeeld lokale commerciële banken versus internationaal gediversifieerde financiële conglomeraten) als de aard van de risico's (bijvoorbeeld operationeel risico versus marktrisico). Als gevolg hiervan verschuiven de internationale regelgevers en toezichthouders van een zuiver regelgebaseerde aanpak naar een marktgebaseerde aanpak, waarbij het bevoegde banken mits goedkeuring toegestaan is hun eigen intern risicobeheersysteem te gebruiken om de voor hen optimale kapitaalreserve te berekenen [45].

6.2.1 Marktgebaseerde risicoanalyse

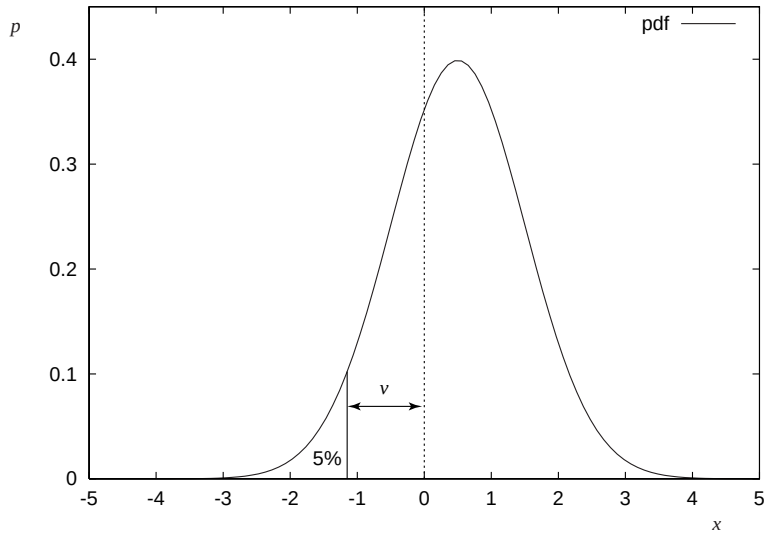
Onder *marktrisico* (*E.* market risk) worden de verliezen verstaan ten gevolge van ongunstige ontwikkelingen in marktkoersen (bijvoorbeeld aandelen) en marktinteresten (bijvoorbeeld interestkoersen en wisselkoersen). Voor het schatten van dit marktrisico maken zowel financiële instellingen als niet-financiële bedrijven veelvuldig gebruik van value-at-risk-modellen. De *value-at-risk* of VaR is een samenvattende numerieke statistische maat voor het mogelijk portefeuilleverlies onder normale marktomstandigheden [67]. Verliezen groter dan de value-at-risk worden enkel met een voorgedefinieerde waarschijnlijkheid geleden, waarbij een specifieke probabilistische verdeling van de relevante marktveranderlijken verondersteld wordt.

De value-at-risk is een eenvoudige grootheid en heeft van nature uit de aantrekkelijk eigenschap dat het een consistente maat voorstelt voor het risico tegenover een groot aantal uiteenlopende posities en risicofactoren. Hierbij is er inherent een correlatiestructuur tussen de risicofactoren aanwezig. Omdat de methodologie van de value-at-risk het maximum bedrag schat dat verloren kan worden overeenkomstig een voorgedefinieerd betrouwbaarheidsinterval en een voorgedefinieerde tijdsduur, kan de voorspelling van de value-at-risk gebruikt worden om de kapitaalvereisten op te stellen op het niveau van de globale instelling. Omdat er met het beschikbaar houden van kapitaal een kost geassocieerd is, is het van cruciaal belang dat de voorspelling van de value-at-risk zo precies mogelijk verloopt.

6.2.2 Value-at-risk als statistische maat

De reeks van de dagelijkse rendementen $\{x_t\}$ van een financieel instrument kan beschouwd worden als één uitkomst van een onderliggend toevalsproces $\{X_t\}$. Onder normale omstandigheden is volledige en gedetailleerde kennis van dit toevalsproces van fundamenteel belang om de toekomstige evolutie te voorspellen en het geassocieerde risico te kwantificeren. Helaas is slechts één uitkomst van dit proces gekend en moeten een aantal eigenschappen zoals stationariteit (of tenminste quasi-stationariteit) en ergodiciteit aangenomen worden om zinvolle voorspellingen te kunnen maken.

Strikt genomen wordt de *value-at-risk* v gedefinieerd als het maximale bedrag dat verloren kan worden volgens een waarschijnlijkheid p_v en over een horizon h, formeel $\Pr[(X_{t+h} - X_t) < -v] = p_v$. Deze waarde kan geschat worden door eerst de onderliggende voorwaardelijke waarschijnlijkheidsdichtheidsfunctie $p(x_{t+h}|\Omega_t)$ te schatten, waarbij Ω_t alle informatie uit het heden en het verleden voorstelt. Figuur 6.1 illustreert de interpretatie van de value-at-risk v voor een geschatte pdf voor X_{t+h}. Typische waarden voor de probabi-



Figuur 6.1: Definitie van de value-at-risk v. De probabiliteit dat het verlies over een horizon h groter is dan v bedraagt precies p_v. In het voorbeeld van de figuur is $p_v = 0.05$ en is de pdf gaussiaans met $\mu = 0.5$ en $\sigma = 1$.

liteit p_v zijn 1% en 5%, terwijl de horizon h typisch 1, 5 of 25 dagen bedraagt, wat respectievelijk overeenkomt met een dag, een week en een maand.

Het schatten van de value-at-risk wordt aanzienlijk vereenvoudigd indien een gaussiaanse verdeling verondersteld wordt, een aanpak die uitvoerig beschreven wordt door RiskMetrics [114]. Het volstaat de verwachtingswaarde μ en de variantie σ^2 te schatten en de gehele pdf is hiermee gekend. Er kan eenvoudig aangetoond worden dat de value-at-risk onder deze voorwaarden op het teken na gegeven wordt door $v = \mu - \alpha\sigma$, waarbij $\alpha = 1.65$ en $\alpha = 2.33$ voor respectievelijk $p_v = 0.05$ en $p_v = 0.01$. Deze aanpak levert behoorlijk consistente resultaten op, maar heeft als belangrijk nadeel dat de geobserveerde dikke staarten niet correct in rekening gebracht worden en dat er geen voor de hand liggende manier is om de niet-stationariteit in de gegevens te schatten. Onder *dikke staarten* (*E.* fat tails) verstaat men de relatief hogere waarschijnlijkheid om extreme waarden waar te nemen ten opzichte van een gaussiaanse distributie met eenzelfde variantie.

6.3 Financieel contextmodel

Het gebruik van contextmodellering om niet één gezamenlijke maar meerdere contextgebaseerde lichtjes afwijkende pdf's te schatten is intuïtief aantrekkelijk. De contexten zijn hierbij gebaseerd op de waarden van een aantal voorafgedefinieerde *priors* (*E.* priors) waarvan theoretisch of empirisch aangetoond is dat ze een voorspellende kracht hebben. Voorbeelden voor priors zijn het historisch verloop van het financiële instrument, interestkoersen, wisselkoersen en zakencyclische voorwaarden. Elke context beschrijft een bepaalde combinatie van priors en kan geïnterpreteerd worden als een gekwantiseerde "toestand van de wereld". Marktgegevens uit het verleden worden gebruikt om de eigenlijke contexten vast te leggen. Nieuwe observaties van de priors leiden automatisch tot de identificatie van een specifieke context en een specifieke pdf horend bij deze context.

6.3.1 Financiële modellering versus datacompressie

Deze vorm van statistische voorspelling, waarbij voor elke context de gehele pdf geschat wordt eerder dan de verwachtingswaarde, ligt precies aan de basis van de gelijkaardige modelvorming als bij datacompressie. De meest geavanceerde technieken voor datacompressie zijn eveneens statistisch van aard en de geschatte pdf wordt er gebruikt om een entropiecoder aan te sturen om de gecomprimeerde bitstroom te genereren [14, 215].

Naast de parallellen met contextmodellering voor compressie, zijn er ook een aantal essentiële tegenstellingen.

- Een eerste tegenstelling ligt in de beschikbaarheid van bronwaarden waaruit het model opgebouwd wordt. Daar waar er voor datacompressie doorgaans heel veel bronwaarden voorhanden zijn (grootteorde miljoenen), is dit aantal voor financiële modellering aanzienlijk kleiner (grootteorde honderden tot duizenden). Als gevolg hiervan is het risico op datagappen heel groot geworden. *Datagappen* (*E.* data snooping) betekent dat informatie bekomen uit de observaties vervolgens aangewend wordt om dezelfde data te onderzoeken [26]. Het leidt tot een uitstekende aanpassing binnen het bereik van de waargenomen gegevens, maar tevens ook tot buitensporige voorspellingen in de toekomst.

- Een tweede tegenstelling is dat voor datacompressie de gehele pdf zo accuraat mogelijk geschat moet worden, terwijl voor risicoanalyse enkel de correctheid van het model ter hoogte van de staarten van belang is. Een goede schatting voor extreem gedrag vereist een hoger aantal

bronwaarden om statistieken op te stellen [36]. Dit vormt een tweede factor die het risico op datagappen verhoogt.

- Een derde tegenstelling is dat het contextmodel niet zuiver causaal en reversibel moet zijn. Dit brengt met zich mee dat de niet-gehele natuur van de gegevens bij financiële analyse geen drempel vormt voor de modellering en dat er ook externe priors kunnen gebruikt worden.

- Een vierde tegenstelling tenslotte ligt in de computationele vereisten van de implementatie van het model. Daar waar een compressietechniek voldoende snel en geheugenefficiënt moet zijn om in de praktijk gebruikt te worden, vervallen deze vereisten nagenoeg volledig voor financiële modellering.

Deze tegenstellingen hebben als gevolg dat een praktische implementatie van een contextmodel voor financiële modellering aanzienlijk zal verschillen van een contextmodel voor datacompressie.

In dit hoofdstuk stellen we een financieel contextmodel voor dat de dynamiek van het marktrisco geassocieerd met de Amerikaanse aandelenmarkt moet onderscheppen. Een venster op het verleden dat enkele duizenden dagelijkse rendementen telt, wordt als datatraining gebruikt om de contexten op te bouwen en de pdf's te schatten. Eenmaal de contexten gedefinieerd zijn, kan de "huidige toestand van de wereld" geschat worden. De bijhorende pdf en de daaruit volgende value-at-risk worden afgeleid uit de observaties uit het verleden horend bij dezelfde toestand.

De analyse wordt uitgevoerd op een tijdsinterval van één dag als elementaire tijdsstap. De keuze van de horizon voor de voorspelling is enigszins arbitrair, maar de dagelijkse frequentie is een redelijke keuze omdat er kan aangenomen worden dat de herverdeling van de aandelenportefeuille bij actieve marktdeelnemers een gelijkaardig patroon volgt [228]. Daarenboven vereisen financiële toezichthouders dat banken hun value-at-risk op een dagelijkse basis herberekenen omdat de hoge liquiditeit van de Amerikaanse aandelen- en derivatenmarkten toelaat dat investeerders hun risicovolle posities sneller sluiten. Intra-dag-analyses hebben het potentieel om het probleem van datagappen in zekere mate te verminderen, maar de noodzakelijke waarnemingen daarvoor ontbreken over een voldoende lange termijn.

6.3.2 Contextmodel en priors

De tegenstellingen tussen de vereisten voor een contextmodel voor datacompressie en een contextmodel voor value-at-risk-voorspelling brengen een aantal wijzigingen met zich mee. Enerzijds is de contextgeneratie wezenlijk an-

ders en anderzijds neemt ook de voorstelling van de eigenlijke pdf een andere vorm aan.

Priors

De eigenschap dat conditioneren de entropie verlaagt ligt aan de grondslag van het contextmodel. Het conditioneren van een toevalsveranderlijke op een andere toevalsveranderlijke die er niet onafhankelijk van is, zal de entropie doen afnemen en bijgevolg neemt de efficiëntie van de modelvorming toe.

Bij financiële modellering is het mogelijk om naast de interne toevalsgrootheid X uit het verleden ook externe toevalsgrootheden Z te gebruiken als *prior* (*E.* prior) voor de contextgeneratie. Deze omvat zowel micro-economische als macro-economische grootheden waarvan aangenomen wordt dat ze een voorspellende kracht hebben. De context $\mathbf{z}_t$ wordt een reële vector waarbij elke component z_t^i een zorgvuldig geselecteerde prior voorstelt.

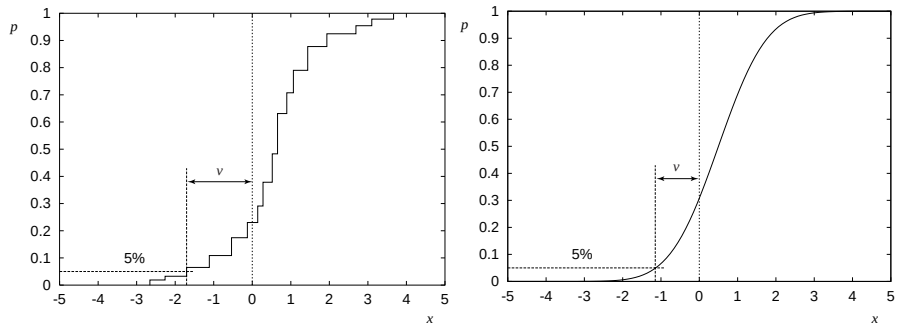
Omdat de contextruimte een continue hogerdimensionale ruimte $\mathcal{Z}$ voorstelt is de kans heel klein dat een bepaalde context $\mathbf{z} \in \mathcal{Z}$ meer dan eens voorkomt. Het probleem van contextverdunning is inherent aanwezig en daarom is een additionele bewerking nodig die de continue context $\mathbf{z} \in \mathcal{Z}$ afbeeldt op een contextklasse $y \subset \mathcal{Z}$. Deze afbeelding stelt een partitionering voor van de contextruimte $\mathcal{Z}$ en wordt de *contextafbeeldingsfunctie* (*E.* context mapping function) genoemd. De verzameling van alle contextklasses y wordt voorgesteld als $\mathcal{Y}$. Elke contextklasse stemt overeen met de eerder gehanteerde term "toestand van de wereld".

Het contextmodel schat de conditionele pdf $p(x_{t+h}|y_t)$ op basis van alle eerder waargenomen evoluties horend bij dezelfde contextklasse. De voorstelling van de pdf kan al dan niet parametrisch zijn.

Niet-parametrisch probabiliteitsmodel

Een niet-parametrische voorstelling van het probabiliteitsmodel houdt een lijst bij van historische waarnemingen uit het verleden. Uit deze waarnemingen volgt een schatting voor de cumulatieve distributiefunctie die vervolgens de value-at-risk v voor een gegeven probabiliteit p_v voorspelt. Omdat p_v vrij klein is moeten er voldoende waarnemingen in rekening gebracht worden om een betrouwbare voorspelling van v te bekomen. Bovendien moet er een vorm van interpolatie toegepast worden. De grafiek links in figuur 6.2 illustreert dit probabiliteitsmodel. Deze aanpak wordt ook wel de *historische* aanpak genoemd.

In combinatie met contextmodellering betekent dit dat elk van de historische waarnemingen (y_u, x_{u+h}) uit het verleden $u = 0 \,..\, t$ toegewezen wordt aan



Figuur 6.2: Het probabiliteitsmodel kan niet-parametrisch (*links*, ook wel historisch genoemd) of parametrisch voorgesteld worden (*rechts*). De schatting van de value-at-risk v voor een gegeven probabiliteit $p_v = 0.05$ loopt in beide gevallen verschillend.

de bijhorende contextklasse y_u. Bij elke contextklasse $y \in \mathcal{Y}$ hoort een onafhankelijke cumulatieve distributiefunctie en bijhorende geschatte value-at-risk. Omdat de onderliggende toevalsveranderlijken zelfs binnen een contextklasse niet volledig stationair zijn, worden de waarnemingen gewogen met een tijdsafhankelijke factor zodat het belang van oudere waarnemingen afneemt.

De sterkte van de historische modellering zit in het beperkt aantal veronderstellingen: behalve stationariteit en ergodiciteit worden geen expliciete veronderstellingen gemaakt betreffende de probabiliteitsdichtheidsfunctie. Zolang er maar voldoende waarnemingen beschikbaar zijn, is het mogelijk de value-at-risk v arbitrair nauwkeurig te schatten. Maar in het voorhanden zijn van waarnemingen schuilt precies ook de zwakte van deze aanpak. A fortiori voor het voorspellen van extreem gedrag loopt het vereiste minimum aantal waarnemingen per contextklasse al snel op tot grootteorde honderd. Wat op zijn beurt betekent dat het aantal contextklassen heel laag moet gehouden worden.

Parametrisch probabiliteitsmodel

Een parametrische voorstelling van het probabiliteitsmodel gaat uit van een vooropgestelde klasse dichtheidsfuncties en enkel die parameters die de dichtheidsfuncties onderscheiden binnen die klasse worden geschat. Zo is het in financiële modellering gebruikelijk een lognormale verdeling te veronderstellen en enkel de verwachtingswaarde μ en de variantie σ^2 worden geschat op basis van de waarnemingen. Merk op dat een toevalsveranderlijke X *lognormaal* (*E*. lognormal) verdeeld is indien $\log X$ normaal verdeeld is. De grafiek rechts in figuur 6.2 illustreert dit probabiliteitsmodel.

In combinatie met contextmodellering betekent een parametrisch probabiliteitsmodel dat voor elke contextklasse $y \in \mathcal{Y}$ een onafhankelijk stel parameters geschat wordt op basis van waarnemingen horend bij de contextklasse y. Ook hier kan wegen met een tijdsafhankelijke factor helpen om de invloed van de niet-stationariteit binnen een context te onderdrukken.

Analyse en beperkingen

Er zijn zeker en vast aanwijzingingen dat historisch gedrag toelaat om voorspellingen te maken in de toekomst [27]. Toch zijn er een aantal tekortkomingen geassocieerd aan het toepassen van probabiliteitsmodellen op werkelijke waarnemingen. Ten eerste gaat de veronderstelling van stationariteit voor veel types van financiële gegevens niet op [3]. Onderzoek naar het verband tussen volatiliteit en regimewijzigingen in tijdsreeksmodellen lijkt aan te geven dat de waargenomen niet-stationariteit verklaard kan worden door *volatiliteitsclusters*, dit zijn tijdsintervallen van verhoogde volatiliteit afgewisseld door tijdsintervallen van normale volatiliteit [3, 242]. In de financiële literatuur bieden conditionele volatiliteitsmodellen en *keerpuntmodellen* (*E.* change-point models) een oplossing voor deze tekortkoming. Het is dan ook gebruikelijk om rendementen te modelleren als een mengeling van verdelingen die elk afzonderlijk conditioneelnormaal verdeeld zijn [26].

Een eerste stap om dit intrinsiek probleem aan te pakken is om de individuele absolute prijsreeksen $\{R_t\}$ om te zetten in een equivalente reeks waarden met nagenoeg tijdsonafhankelijk bereik. Om de berekening van opeenvolgende prijsreeksverschillen te vereenvoudigen zijn vooral *continue samengestelde rendementen* (*E.* continuously compounded returns) $X_t = \log(R_t/R_{t-1})$ heel effectief. Deze grootheid wordt ook wel als het *logrendement* (*E.* log return) omschreven. Statistische analyse toont aan dat deze reeksen nog altijd in zekere mate niet-stationair zijn. Om deze reden wordt iedere historische waarneming vermenigvuldigd met een gewicht $w(\delta_t)$, waarbij $w(\delta_t)$ een monotoon dalende functie is van het tijdsverschil δ_t tussen de tijd van de historische waarneming en de huidige tijd.

Het tijdsverschil δ_t kan absoluut of relatief gemeten worden. Bij een absolute tijdsmeting is δ_t per definitie het rekenkundig verschil tussen beide tijdsindices. Bij een relatieve tijdsmeting worden alle historische waarnemingen horend bij de huidige contextklasse chronologisch gerangschikt en is δ_t gedefinieerd als het verschil in rangorde-index tussen de historische waarneming en de huidige waarneming. Stel bijvoorbeeld dat de referentiewaarneming zich 10 dagen geleden voordeed, maar dat het de vorige historische waarneming is binnen de huidige context. Dan is $\delta_t = 10$ bij absolute tijdsmeting en is $\delta_t = 1$

bij relatieve tijdsmeting. Doorgaans neemt de tijdswegingsfunctie $w(\delta_t)$ een exponentieel dalende vorm aan, dus $w(\delta_t) = \lambda^{\delta_t}$ met $0 < \lambda \leq 1$.

Daarenboven zijn de contextafhankelijke verdelingen in het voorgestelde model geconditioneerd op een grootheid die zelf een toevalskarakter vertoont en op zijn beurt gebaseerd is op statistieken van de te modelleren grootheid. Als gevolg van deze interne terugkoppeling zijn de contextklassen, die als identificatie van de toestand van de wereld dienen en waarbinnen het verwachte rendement en volatiliteit als constant kan beschouwd worden, *endogeen* of "van binnenuit" gegenereerd. Deze vorm van terugkoppeling fungeert als compensatie voor eventuele niet-stationariteiten en is te verkiezen boven de aanpak waarbij nagegaan wordt of de individuele rendementreeksen stationair zijn wat betreft verwachtingswaarde en variantie.

Samengevat betekent dit dat de intrinsiek aanwezige niet-stationariteit aangepakt wordt door het contextmodel op meerdere manieren adaptief te maken: door naar een equivalente voorstelling $\{X_t\}$ over te gaan, door de historische waarnemingen te wegen volgens hun leeftijd, door waarnemingen onder te brengen in disjuncte contextklassen en door het systeem intern terug te koppelen wat aanleiding geeft tot nieuwe contextklassen getraind met recente gegevens.

Een tweede ernstige tekortkoming van het contextmodel is eigen aan alle modellen die uitsluitend getraind worden met historische waarnemingen. Het model is enkel in staat situaties te herkennen die vroeger al opgetreden zijn. De gevolgen hiervan zijn tweeledig: enerzijds beschouwt het model hoogst onwaarschijnlijke situaties als onmogelijk en anderzijds zal het model, indien een dergelijke onwaarschijnlijke situatie zich ooit heeft voorgedaan en in rekening gebracht wordt, ze onmiddellijk als een typische situatie beschouwen. Vooral in het kader van risicoanalyse legt dit ernstige beperkingen op aan de efficiëntie van de voorspelling van de value-at-risk. Om deze reden hebben we de crash van 1987 verwijderd uit de waarnemingen die als training dienen voor de experimenten.

6.3.3 Boomgestructureerde vectorkwantisatie

Het bepalen van de contextafbeeldingsfunctie dient heel voorzichtig te gebeuren. Er zijn relatief weinig historische waarnemingen beschikbaar maar toch moet de contextclassificatie zo verlopen dat binnen elke klasse voldoende waarnemingen aanwezig zijn om statistisch significante uitspraken over extreem gedrag mogelijk te maken. Om contextverdunning tegen te gaan zal het aantal contextklassen dan ook beperkt zijn. In tegenstelling tot beeldcompressie waar het aantal contextklassen kan oplopen tot in de duizenden of zelfs in

de miljoenen, zal dit aantal voor financiële modellering niet hoger dan maximaal enkele honderden mogen bedragen.

Daar komt bij dat de dimensionaliteit van de contextruimte aanzienlijk kan zijn. Een eenvoudige scheidbare contextafbeeldingsfunctie waarbij elk van de priors afzonderlijk gediscretiseerd wordt, is hierdoor bij voorbaat uitgesloten. Deze aanpak zou immers tot een exponentieel stijgend aantal contextklassen leiden. Stel bij wijze van voorbeeld dat er 4096 historische waarnemingen zijn en dat elk van de $q = 8$ priors opgesplitst wordt in 2 klassen, dan leidt dit contextmodel tot $2^q = 256$ contextklassen met gemiddeld 16 waarnemingen per klasse. Dit is onvoldoende om een value-at-risk te voorspellen voor $p_v = 0.05$ en a fortiori indien $p_v = 0.01$. Deze "vloek van de dimensionaliteit" is een fundamenteel probleem dat vraagt om een intelligent partitiealgoritme om de contextruimte op te splitsen. We stellen hiervoor als oplossing dynamische boomgestructureerde vectorkwantisatie voor.

Tijdens het verwerken van de eerste waarnemingen is het onmogelijk statistisch significante voorspellingen te maken. Om deze reden wordt er eerst een trainingsfase ingelast. Gedurende deze initiële fase worden geen voorspellingen gemaakt, maar ze dient enkel om het model te trainen met initiële statistieken. Na afloop van deze fase gaat het model over op de evaluatiefase, waarbij accurate voorspellingen van de value-at-risk gecombineerd worden met de training op basis van de waarnemingen. Gedurende deze evaluatiefase genereert het model dagelijks een voorspelling voor de value-at-risk over de horizon en hiervoor gebruikt het alle historische waarnemingen tot op heden.

Partitionering van de contextboom

Een eenvoudige schatting geeft een ruw idee van een haalbaar aantal contextklassen. Indien het model getraind wordt met dagelijkse waarnemingen van de voorbije dertig jaar, dan zijn er tussen de 2000 en 8000 trainingskoppels beschikbaar. Om statistisch significante uitspraken te kunnen doen zijn er tenminste 100 tot 200 waarnemingen nodig per contextklasse. Dit betekent dat er maximaal tussen 10 en 40 contextklassen kunnen gecreëerd worden.

Dit probleem wordt opgelost in twee stappen. Ten eerste wordt de contextruimte $\mathcal{Z}$ gepartitioneerd in een aantal contextklassen y. De contextafbeeldingsfunctie beeldt elke context $\mathbf{z} \in \mathcal{Z}$ af op één van die klassen gebaseerd op een kleinsteafstandscriterium tot het massamiddelpunt binnen die klasse, een techniek die gekend staat als vectorkwantisatie [82]. Ten tweede is er nood aan een mechanisme dat deze partitionering beheert en verder laat evolueren naarmate meer waarnemingen beschikbaar zijn. Daarom worden deze contextklassen ondergebracht in een groeiende boomstructuur die kan wijzigen op een

dagelijkse basis. Als de nieuwe contextklassen op een zodanige manier ontstaan dat ze rekening houden met de teruggekoppelde rendementen, dan ligt het in de lijn der verwachting dat de structuur van deze contextclassificatieboom na verloop van tijd verborgen informatie over de voorspellende kracht van gezamenlijke priors zal blootleggen.

Het verwerken van een individuele waarneming x_t bestaat dan ook uit twee stappen. Een eerste stap bepaalt de bijhorende context $\mathbf{z}_t$ en beeldt deze af op een contextklasse y_t. Een tweede stap koppelt de informatie van de gezamenlijke waarneming $(\mathbf{z}_t, x_{t+h})$ terug naar het probabiliteitsmodel. Formeel uitgewerkt zien deze stappen er als volgt uit.

- **contextafbeelding en value-at-risk-schatting**: Voor iedere contextklasse $y \in \mathcal{Y}$ kan een massamiddelpunt $\bar{\mathbf{z}}_y = \sum_{i:\mathbf{z}_i \in y} \mathbf{z}_i / l(y)$ gedefinieerd worden. In eerste instantie wordt de waarde van $\mathbf{z}_t$ bepaald op basis van de individuele priors. Om datagappen te vermijden komt hiervoor enkel informatie in aanmerking die op tijdstip t beschikbaar is, ook en vooral tijdens de trainingsfase. Vervolgens beeldt de contextafbeeldingsfunctie deze context $\mathbf{z}_t$ af op de contextklasse y waarvoor de afstand $||\bar{\mathbf{z}}_y - \mathbf{z}_t||$ minimaal is. De pdf behorend bij deze klasse laat toe de value-at-risk te schatten. Deze pdf kan parametrisch of historisch zijn.

- **terugkoppeling van de waarneming**: De gezamenlijke waarneming $(\mathbf{z}_t, x_{t+h})$ vormt de basis voor alle kennis binnen het model en deze moet dan ook teruggekoppeld worden naar het model op tijdstip $t + h$. Hiertoe wordt de context $\mathbf{z}_t$ toegevoegd aan de contextklasse y_t en wordt een nieuw massamiddelpunt $\bar{\mathbf{z}}_y$ berekend. De pdf horend bij deze contextklasse wordt eveneens aangepast met de waarneming x_{t+h}. Voor een parametrisch probabiliteitsmodel betekent dit dat de parameters herberekend worden. Voor een historisch model wordt de waarneming x_{t+h} toegevoegd aan de lijst van waarnemingen horend bij de contextklasse.

Een gevolg van deze aanpak is dat de partitionering van de contextruimte en de eigenlijke toestand van de wereld horend bij een gegeven contextklasse niet constant is in de tijd. Na het terugkoppelen van de meest recente waarneming controleert het algoritme of de contextboomstructuur moet aangepast worden. Dit gebeurt door een eindknoop van de contextboom op te splitsen in twee nieuwe knopen.

Splitsingsalgoritme

De contextboom bestaat in het begin uit slechts één knoop en de contextclassificatie beeldt alle geobserveerde contexten af op dezelfde initiële context-

klasse. Indien voldaan is aan een voorgedefinieerd *maturiteitscriterium* (*E.* maturity criterion) wordt een knoop gesplitst in een aantal kindknopen, typisch twee. Dit maturiteitscriterium betekent doorgaans dat de geassocieerde contextklasse vaker is voorgekomen dan een voorafgedefinieerde drempelwaarde. De oude knoop wordt een vaderknoop en de geassocieerde waarnemingen worden gesplitst over de kindknopen. Na deze herverdeling wordt de vaderknoop niet langer gebruikt en zijn enkel de kindknopen nog geldig. Het genereren van een aantal kindknopen uit een vaderknoop kan verlopen op een aantal manieren:

- **willekeurige knoopgeneratie**: Met de vaderknoop y is er een massamiddelpunt $\bar{\mathbf{z}}_y$ en een lijst contextwaarnemingen $\{\mathbf{z}_i\}$ geassocieerd. De twee kindknopen worden gegenereerd door een arbitrair kleine stoorvector $\epsilon \in \mathcal{Z}$ respectievelijk bij het massamiddelpunt op te tellen en af te trekken. Op deze manier onstaan twee nieuwe attractoren $\bar{\mathbf{z}} \pm \epsilon$ en elk van de waargenomen contexten van $\{\mathbf{z}_i\}$ wordt toegewezen aan één van beide kindknopen gebaseerd op de dichtste attractor. Het criterium voor deze classificatie is de euclidische afstand en om deze reden wordt elk van de dimensies van de contextruimte vooraf genormaliseerd. Na de herverdeling van de contextwaarnemingen wordt de initiële attractor horend bij elk van de kindknopen vervangen door het effectieve massamiddelpunt.

- **snelle min-max knoopgeneratie**: Het grootste nadeel van de willekeurige knoopgeneratie is dat er geen rekening wordt gehouden met de effectief waargenomen rendementen x_{t+h}. Dit betekent dat de vectorkwantisatie uitsluitend in de contextruimte plaatsvindt zonder rekening te houden met de kennis aanwezig in de gezamenlijke waarnemingen van context en rendement. De snelle min-max knoopgeneratie houdt wel rekening met deze gezamenlijke waarnemingen en de finale boomstructuur kan op deze manier verborgen informatie blootleggen over de voorspellende kracht van de individuele priors. Van alle contexten $\{\mathbf{z}_i\}$ horend bij een gegeven vaderknoop y worden eerst die twee geselecteerd die een extreem rendement x_{i+h} met zich meebrengen. Stellen x^+ en x^- respectievelijk het maximale en de minimale rendement voor en zijn verder $\mathbf{z}^+$ en $\mathbf{z}^-$ de bijhorende contexten, dan dienen deze beide contexten als initiële attractoren voor de generatie van de kindknopen. Net als bij willekeurige knoopgeneratie wordt elke context horend bij de vaderknoop ondergebracht bij één van beide kindknopen volgens een kleinsteafstandscriterium tot de attractor. Na classificatie vervangt het nieuw te berekenen massamiddelpunt de attractor.

- **volledige min-max knoopgeneratie**: De snelle min-max knoopgeneratie heeft meer potentieel dan de willekeurige knoopgeneratie omdat de informatie over de rendementen teruggekoppeld wordt naar het vectorkwantisatieproces. Helaas is deze aanpak heel gevoelig aan extreme waarnemingen en veronderstelt het een zekere mate van monotoniciteit voor het gedrag van de rendementen als functie van de priors. Dit probleem kan vermeden worden door de informatie geassocieerd met alle rendementen te gebruiken om de contexten te classificeren. De volledige min-max knoopgeneratie definieert daartoe een drempelrendement $\hat{x}$ als $(x^+ + x^-)/2$. Elke context $\mathbf{z}_i$ wordt ondergebracht bij één van beide kindknopen gebaseerd op het feit dat het geassocieerde rendement $x_{i+h} > \hat{x}$ dan wel $x_{i+h} < \hat{x}$. Net als bij de vorige splitsingsalgoritmen wordt een nieuw massamiddelpunt gedefinieerd voor elke kindknoop als de herverdeling afgelopen is.

Elk van deze splitsingsalgoritmen vervangt één vaderknoop door twee kindknopen en voegt aldus één contextklasse toe aan de actieve lijst van contextklassen. De algoritmen zijn bovendien onafhankelijk van de precieze dimensionaliteit van de contextruimte.

Additionele verbeteringen

Tot slot worden nog een tweetal additionele verbeteringen voorgesteld voor het contextmodel. Ten eerste is er het "omgekeerd herstarten" van het model dat de gevolgen van de niet-stationariteit van de waarnemingen op de contextgeneratie vermindert. Dit gebeurt door relatief meer waarde te hechten aan de meest recente waarnemingen met betrekking tot het groeien van de contextboom. Ten tweede is er een "terugkoppelingsmechanisme" dat als doel heeft om enerzijds de trainingstijd te verminderen en anderzijds consequente over- of onderschatting te vermijden. Formeel verlopen deze verbeteringen als volgt:

- **omgekeerd herstarten**: Onder normale omstandigheden groeit de contextboom vanaf de historisch oudste waarnemingen en past die zich geleidelijk aan aan de meer recente gebeurtenissen. Omdat de initiële splitsingen de belangrijkste takken van de boom bepaald hebben, betekent dit dat de kern van de boomstructuur gebaseerd is op de oudste waarnemingen. Gelet op de waargenomen niet-stationariteit kan het voordelig zijn indien de belangrijkste takken van de boom precies ontstaan zijn op basis van recente waarnemingen. Dit gebeurt door het contextmodel en de geassocieerde contextboom periodisch te herstarten vertrekkend van de meest recente observaties. Na elke periode t_r wordt de volgorde van

de waarnemingen omgekeerd en de volledige contextboom wordt volledig opnieuw opgebouwd. De meest recente waarnemingen bepalen nu de initiële takken van de boom terwijl de oudste waarnemingen verantwoordelijk zijn voor de verfijnde structuur van de eindknopen. Het weze duidelijk dat de volgorde van de verwerking een compromis is want de meest recente waarnemingen zouden liefst gebruikt worden voor zowel de initiële als voor de verfijnde vertakking.

- **terugkoppelingsmechanisme**: Terugkoppeling van de efficiëntie van de voorspelde value-at-risk kan een enorm effect hebben op de dynamiek van de training. Bovendien kan een consistente misvoorspelling naar boven of beneden toe sneller onderschept en vermeden worden. Het terugkoppelingsmechanisme voegt een artificiële prior toe aan de lijst van economische priors. Deze additionele prior is te interpreteren als een binaire vlag die aangeeft of de meest recente waarneming de voorspelling al dan niet overtrof. Net als de andere priors moet deze veranderlijke genormaliseerd worden alvorens die aan de contextruimte toe te voegen.

De impact van het terugkoppelingsmechanisme op de snelheid is verwaarloosbaar, maar het omgekeerd herstarten kan de modellering aanzienlijk vertragen indien de herstartperiode t_r heel klein is.

6.3.4 Discussie: niet-lineaire modellering

Het voorgestelde model is aanzienlijk niet-lineair maar de aard van de voorspelling verschilt aanzienlijk van die zoals meestal waargenomen in niet-lineaire modellen.

Conventionele niet-lineaire modellen van een gegeven systeem vertrekken van een stelsel van niet-lineaire differentiaalvergelijkingen dat een beperkt aantal parameters en veranderlijken kent. Van nature uit is de tijdsdimensie continu maar voor de meeste praktische implementaties moet die gediscretiseerd worden. De invoer vanuit de geobserveerde realiteit beperkt zich tot al dan niet geschatte parameters en randvoorwaarden, doorgaans is dit de huidige toestand van het systeem. De oplossing van dit systeem is in theorie deterministisch, maar in de praktijk meestal chaotisch op langere termijn. Het stelsel differentiaalvergelijkingen is doorgaans zelf tijdsonafhankelijk en beschrijft de dynamiek van het systeem expliciet en formeel.

Het voorgestelde contextmodel daarentegen is veel algemener omdat er veel meer soorten gedrag kunnen in gevat worden. Het belangrijkste verschil met het conventioneel niet-lineair systeem schuilt in de stochastische natuur van de aanpak. Het is inherent aan de modelvorming dat meerdere uitkomsten mogelijk zijn en niet zozeer de uitkomsten maar de probabiliteit van deze

uitkomsten wordt geschat op statistische basis. De tijdsveranderlijke is discreet en de invoer vanuit de geobserveerde realiteit is veel groter omdat alle informatie betreffende het systeem afkomstig is van training. Enkel een beperkt aantal veronderstellingen betreffende stationariteit en continuïteit van de pdf worden voorafgaandelijk gemaakt. De rendementen worden beschreven als een mengeling van een beperkt aantal onafhankelijke stationaire bronnen. De parameters van het model worden geoptimaliseerd door een exhaustieve zoektocht, een proces dat gevoelig is voor datagappen. Het contextmodel is tijdsafhankelijk en beschrijft de dynamiek van het systeem op een impliciete manier. Omdat zoveel verschillende gedragstypes kunnen gemodelleerd worden en omdat zo weinig veronderstellingen gemaakt worden, heeft het model grote aantallen observaties uit de reële wereld nodig om accurate voorspellingen te maken. Het trainen is gelijkaardig aan markovmodellering, maar de aanpak verschilt omdat het model geen interne toestandstransities schat maar externe informatie van de priors gebruikt om precies de toestanden zelf vast te leggen.

6.4 Evaluatietechnieken

Het evalueren van de nauwkeurigheid van de voorspellingen voor de value-at-risk is geen evidente taak. In de financiële wereld eisen de toezichthoudende autoriteiten dat de schatting van de value-at-risk die volgt uit het intern risicobeheersysteem van een bank vermenigvuldigd wordt met een veiligheidsfactor om het minimaal vereiste kapitaal te bepalen. De standaardmethode voor banken om hun nauwkeurigheid te evalueren is om het aantal observaties te tellen die de voorspelde value-at-risk overschrijden over een tijdspanne van 250 handelsdagen. Daarnaast zijn een aantal andere methoden voorgesteld die ondermeer proberen gecompliceerde verliesfuncties te minimaliseren. Een belangrijke vraag in de financiële wereld is of het voorgestelde model in staat is om periodes van hogere en lagere volatiliteit te voorspellen. Een hogere volatiliteit betekent immers dat de hoeveelheid gereserveerd kapitaal moet toenemen.

Niettegenstaande elke evaluatie van de schatting van de value-at-risk moeilijk is omwille van het beperkte vermogen van de test, zijn er recent een aantal verbeterde technieken voorgesteld [140, 141]. Om de experimentele simulaties te evalueren stellen we drie types evaluatiematen voor: statistieken betreffende de trefverhouding, een criterium gebaseerd op de χ^2-afstand met betrekking tot de binomiale verdeling en verlies- en kostfuncties. Daartoe dienen eerst een aantal grootheden gedefinieerd te worden.

Vooreerst is er het binair toevalsproces $\{I_t\}$ dat een *treffer* (*E.* hit) genoemd wordt. Hierbij is $I_t = 1$ indien het verlies X_t de voorspelde value-

at-risk v_t overtreft en is $I_t = 0$ in het andere geval. Deze toevalsgrootheid kan geïnterpreteerd worden als een priorfunctie die buitengewoon grote verliezen vlagt, overeenkomstig een voorgedefinieerde betrouwbaarheid p_v. Als het statistisch model erin slaagt alle afwijkingen van het normaal gedrag te onderscheppen en perfect de geobserveerde gegevens kan verklaren, dan is I_t voor elke t een binaire toevalsveranderlijke die de waarde 1 aanneemt met waarschijnlijkheid p_v en de waarde 0 met waarschijnlijkheid $1 - p_v$.

Bevat de gehele evaluatieperiode precies n waarnemingen, dan wordt die opgedeeld in q niet-overlappende vensters van elk l waarnemingen. Voor ieder venster i is de toevalsgrootheid T_i gedefnieerd als $T_i = \sum_{k=0}^{l-1} I_{il+k}$ en indien een perfecte modellering bekomen wordt is de verwachtingswaarde van deze toevalsveranderlijke precies $p_v l$. Bovendien is de reeks toevalsveranderlijken $\{T_i\}$ dan onafhankelijk en identiek verdeeld en voldoet T_i aan de binomiale verdeling

$$\Pr[T_i = j] = \left(\begin{array}{c} l \\ j \end{array} \right) p_v^j (1 - p_v)^{l-j}.$$

Deze toevalsgrootheden liggen aan de basis van de constructie van drie stellen evaluatiecriteria: trefstatistieken, de χ^2-statistiek en verlies- en kostfuncties.

Trefstatistieken

Het eerste stel evaluatiecriteria baseert zich op de waarnemingen voor T_i. Het waargenomen minimum $m^- = \min_i\{T_i\}$, het waargenomen gemiddelde $\bar{m} = \sum_i T_i/q$ en het waargenomen maximum $m^+ = \max_i\{T_i\}$ vormen drie interessante statistieken. Hun verdeling wordt gegeven door:

$$\begin{aligned}
\Pr[m^- < j] &= 1 - \left(\sum_{k=j}^{l} \Pr[T = k] \right)^q, \\
\Pr[q\bar{m} = j] &= \left(\begin{array}{c} ql \\ j \end{array} \right) p_v^j (1 - p_v)^{ql-j}, \\
\Pr[m^+ \geq j] &= 1 - \left(\sum_{k=0}^{j-1} \Pr[T = k] \right)^q.
\end{aligned}$$

De eigenschap *heteroskedasticiteit* (*E.* heteroskedasticity) wijst erop dat de lokale variantie tijdsafhankelijk is. In de ideale situatie, waarbij de heteroskedasticiteit van de rendementen volledig onderschept wordt door het model, moet het waargenomen gemiddelde $\bar{m}$ gelijk zijn aan $p_v l$ en moet het waargenomen maximum m^+ niet te groot zijn. Voor lage waarden van p_v is het waargenomen minimum m^- een nutteloze statistiek.

De χ^2-statistiek

De hypothese dat de waargenomen veranderlijke T_i voldoet aan de binomiale verdeling kan gevalideerd worden aan de hand van Pearsons χ^2-statistiek [78, 226]. Omdat T_i waarden kan aannemen in het interval $[0, l]$, wordt de statistiek van de χ^2-test gegeven door

$$\chi^2 = \sum_{k=0}^{l} \frac{(n_k - e_k)^2}{e_k},$$

waarbij n_k en e_k respectievelijk het geobserveerde en het verwachte aantal vensters voorstelt waar $T_i = k$, overeenkomstig de binomiaalverdeling. Deze statistiek heeft $\nu = l$ vrijheidsgraden.

Verlies- en kostfuncties

Bovenstaande criteria maken enkel gebruik van het aantal treffers en niet van de grootte van de treffer. Een artificiële value-at-risk voor $p_v = 0.01$ die gedefinieerd is als $+\infty$ op elke eerste dag en $-\infty$ voor de overige 99 dagen zou een perfecte score behalen op deze criteria, maar voldoet absoluut niet aan de vereisten van een financieel opportune value-at-risk. Gebaseerd op het idee van regelgevende verliesfuncties voegen we daarom een verlies- en een kostfunctie toe aan de stellen evaluatiecriteria [141]. Deze zijn respectievelijk gebaseerd op de buitengewone waarnemingen (de treffers) en de reguliere waarnemingen.

De *verliesfunctie* (*E.* loss function) L is gedefinieerd als

$$L = \sum_{t=1}^{n} H(v_t - x_t)(v_t - x_t)^2,$$

waarbij de value-at-risk v_t een absolute voorspelling voor het rendement voorstelt (niet langer het voorspelde verlies) en waarbij x_t het waargenomen rendement voorstelt. Verder stelt $H(x)$ de Heaviside-functie voor, gedefinieerd als 1 voor $x > 0$, als $1/2$ voor $x = 0$ en als 0 voor $x < 0$. Een verliesfunctie L' uit de literatuur is gelijkaardig, maar telt er het aantal treffers bij op, zodat $L' = L + \sum_t I_t = L + q\bar{m}$ volgens [141]. Beide verliesfuncties zijn enkel gebaseerd op buitengewone waaarnemingen waarbij het waargenomen verlies het voorspelde verlies overtreft, dus $x_t < v_t$. De verliesfunctie L is een maat voor het verlies dat veroorzaakt wordt door het onderschatten van het risicokapitaal.

De *kostfunctie* (*E.* cost function) C is complementair gedefinieerd als

$$C = \sum_{t=1}^{n} H(x_t - v_t)(x_t - v_t),$$

waarbij andermaal v_t en x_t respectievelijk het voorspelde en het waargenomen rendement voorstellen. De kwadratische vorm is vervangen door een lineaire functie en hiervoor worden enkel de reguliere waarnemingen in rekening gebracht, dus $x_t > v_t$. Het stelt de kost voor die veroorzaakt wordt door het overschatten van het risicokapitaal.

Noch de verliesfunctie noch de kostfunctie mogen als een zelfstandig criterium geïnterpreteerd worden. Ze moeten samen geëvalueerd worden, liefst nog in combinatie met de voorgaande criteria. Onverwacht hoge waarden leren iets over het overdreven over- of onderschatten van het risicokapitaal. Afhankelijk van haar belangen kan een financiële instelling opteren om verschillende gewichten toe te kennen aan de respectieve functies.

6.5 Experimentele resultaten

Het voorgestelde contextmodel is geïmplementeerd in C++ en gevalideerd op zowel Microsoft Windows NT als op een Linux Pentium III besturingssysteem. Afhankelijk van de geselecteerde opties en parameters van het model vergt het verwerken van 8000 waarnemingen tussen 10 en 120 seconden. De huidige implementatie heeft hier 2 megabyte geheugen voor nodig.

6.5.1 Financiële gegevens

De modellering is toegepast op de dagelijkse noteringen van de Amerikaanse Standard&Poor 500 (S&P500) aandelenindex van oktober 1969 tot december 1999. Deze index bevat de aandelen van de grootste bedrijven en vertegenwoordigt hiermee een groot deel van de totale marktkapitalisatie van de New York Stock Exchange. Er kan dan ook redelijkerwijze aangenomen worden dat deze index het geassocieerde marktrisico grotendeels onderschept.

Er zijn in totaal 8089 waarnemingen beschikbaar voor training en evaluatie. De eerste 200 waarnemingen komen niet in aanmerking voor training omwille van een aantal initialisatievoorwaarden, ondermeer omdat eerst een zinvolle waarde moet toegekend worden aan lange-termijn-priors. Waarnemingen 201 tot 2000 worden uitsluitend voor training gebruikt. Dit komt overeen met de periode van 7 oktober 1969 tot 30 augustus 1976. Waarnemingen 2001 tot 8089 worden zowel voor training als voor validatie gebruikt. Dit komt overeen met de periode van 31 augustus 1977 tot 31 december 1999. Het spreekt voor zich dat omwille van causaliteitsredenen enkel waarnemingen uit het verleden in aanmerking komen voor een voorspelling naar de toekomst toe. Helaas zijn niet alle priors beschikbaar vanaf het begin van de trainingsperiode; deze worden vervangen door de waarde 0 wat een licht vertekend beeld kan opleveren.

Tabel 6.1: Overzicht van de priors voor het contextmodel. Er worden zowel technische factoren (z_1, z_2, z_3 en z_4) als fundamentele factoren (z_5, z_6 en z_7) gebruikt.

symbool	prior
z_1	rendement op 1 dag
z_2	rendement op 5 dagen
z_3	rendement op 25 dagen
z_4	volatiliteit op 100 dagen
z_5	differentiële termijnstructuur
z_6	differentiële falingsmarge
z_7	differentieel dividendrendement

In totaal zijn er $n = 6089$ waarnemingen beschikbaar voor evaluatie. Voor een venstergrootte van $l = 100$ waarnemingen komt dit overeen met $q = 60$ niet-overlappende vensters.

Om de trainingsproblemen die voortvloeien uit hoogst onwaarschijnlijke waarnemingen te vermijden, is een volatiele periode van 100 waarnemingen omstreeks de crash van 1987 voor de training verwijderd uit de historische gegevens. Om hun impact alsnog te analyseren, beschrijven we in de laatste paragraaf een aantal resultaten voor de situatie waarin deze volatiele periode wel in rekening gebracht wordt.

Er komen meerdere priors in aanmerking om de contexten op te bouwen. De keuze van de priors is gebaseerd op theoretische modellen en empirische bevindingen beschreven in het onderzoek naar de marktkoersen van activa. Een aantal studies naar de invloeden op deze noteringen tonen aan dat de aandelenkoersen door zowel technische als fundamentele factoren gestuurd worden [80, 90]. Tabel 6.1 geeft een overzicht van de zeven priors die samen de contextruimte vastleggen. De volgende paragrafen beschrijven deze priors en hun voorspellende kracht in detail.

Technische factoren

Als eerste zijn er aanwijzingen gevonden voor de nawerking van dagelijkse rendementen op korte termijn en voor de omkering van het gemiddelde op middellange termijn [26]. Wij gebruiken vier technische veranderlijken om deze effecten te onderscheppen en definiëren ze als de *tendenspriors* (*E*. momentum priors). De eerste drie technische veranderlijken z_1, z_2 en z_3 hebben als doel de dynamiek op korte termijn voor te stellen en ze bevatten het voorbije nettorendement over een periode van respectievelijk één dag, één week en

één maand. De vierde technische veranderlijke drukt de mate van verwachte volatiliteit uit op een gegeven moment, gemeten als de spreiding van de aandelenrendementen over de voorbije 100 marktdagen. Deze contextveranderlijke z_4 wordt berekend als de verhouding van het verschil tussen het maximale en het minimale niveau van de index over de voorbije 100 dagen ten opzichte van het minimale niveau van de index.

Fundamentele factoren

De tweede set priors waarvan verwacht wordt dat ze informatie bevatten over de toekomstige rendementen zijn de macro-economische factoren. Deze fundamentele veranderlijken zijn dikwijls het onderwerp van onderzoek naar multi-factor-modellen en daaruit blijkt dat ze een voorspellende kracht hebben [34, 76].

- De eerste macro-economische veranderlijke is de dagelijkse verandering in de *Amerikaanse rentecurve* (*E.* US yield curve). Deze *factor interestmarge* (*E.* term spread factor) wordt gemeten als het verschil tussen de risicoloze interestkoers op lange termijn en die op korte termijn. Deze worden respectievelijk vervangen door de koers op de Amerikaanse referentieoverheidsobligatie op 10 jaar en het Amerikaans schatkistcertificaat op 3 maanden. Onderzoek wijst uit dat de helling van de termijnstructuur informatie bevat over toekomstige economische groei [88]. Verder onderzoek toont een direct verband aan tussen de termijnstructuur van interestkoersen en buitengewone rendementen op financiële markten [25]. Als gevolg hiervan beïnvloeden wijzigingen in de rentecurve de verwachte aandelenrendementen, al hangt de richting en de grootte van dit effect af van de bron in de wijziging van de termijnstructuur, meer bepaald of de verandering veroorzaakt was door variaties in de interestkoers op korte dan wel op lange termijn. Niet zozeer deze veranderlijke zelf maar het dagelijks verschil wordt als prior z_5 gebruikt voor de contextgeneratie.

- De tweede macro-economische veranderlijke is de *falingsmarge in interestvoeten* (*E.* default spread), bedoeld om de altijd aanwezige invloed van het standaardrisico over alle economische takken heen op de financiële markten te onderscheppen. Theoretisch beschouwd zou een toename in het verwachte rampspoedrisico van ondernemingen tot een toename moeten leiden voor de vereiste rendementen op aandelen. We meten de falingsmarge in de interestvoeten als het verschil tussen het rendement op een ondernemingsobligatie en de risicoloze interestkoers.

Hiervoor komt respectievelijk het rendement op de Amerikaanse referentieondernemingsobligatie volgens de BAA-notering en de koers van de Amerikaanse overheidsobligatie op 10 jaar in de plaats. Merk op dat dit verschil het verwachte standaardrisico uitdrukt omdat de Amerikaanse overheid ook een AAA-notering kent. Net als bij het berekenen van de factor interestmarge wordt het dagelijks verschil z_6 van deze veranderlijke als prior gebruikt.

- De derde macro-economische veranderlijke tenslotte wordt gegeven door het *dividendrendement* (*E.* dividend yield) [25, 26]. De dagelijkse verschillen z_7 in deze reeks laten toe om de verwachtingen te onderscheppen van de investeerders omtrent het uitbetalen van dividenden in de Amerikaanse aandelenmarkt. In theorie zou een toename in het dividendrendement moeten duiden op toegenomen winsten.

Voor de fundamentele factoren komen enkel differentiële waarden in aanmerking als prior. Strikt genomen zou het ook mogelijk zijn om beide termen van het verschil te gebruiken voor het contextmodel. De verschillen hebben als voordeel dat ze een hogere mate van stationariteit vertonen en dat ze bovendien minder trainingswaarnemingen vereisen omdat de vloek van de dimensionaliteit zich minder laat gelden.

6.5.2 Modelparameters

Zoals uitvoerig besproken in dit hoofdstuk maakt het boomgestructureerde contextmodel gebruik van een groot aantal parameters. Een aanvaardbare combinatie voor deze parameters moet empirisch afgeleid worden op basis van de historische waarnemingen.

Tabel 6.2 geeft een overzicht van de parameters die samen de modelvorming vastleggen. De "maturiteitsdrempel" n_m bepaalt het maturiteitscriterium: een knoop zal zich splitsen in een aantal nieuwe knopen van zodra de bezetting deze drempelwaarde overtreft. Het "aantal kindknopen" n_c geeft aan hoeveel nieuwe knopen gecreëerd worden wanneer een eindknoop voldoet aan het maturiteitscriterium en zich splitst. Merk op dat de vorige eindknoop bij dit proces geannihileerd wordt zodat de netto toename in aantal knopen $n_c - 1$ bedraagt. Het "splitsingsalgoritme" A beschrijft de knoopgeneratie en geeft weer hoe de historische waarnemingen toegewezen worden aan de nieuwe eindknopen. Zoals hierboven beschreven kan deze parameter drie waarden aannemen: willekeurige knoopgeneratie, snelle min-max knoopgeneratie en volledige knoopgeneratie. De "tijdsmeting" T_w geeft de manier weer waarop de verlopen tijd tot een referentiewaarneming gemeten wordt. Zoals eerder

Tabel 6.2: Overzicht en korte beschrijving van de modelparameters. De kleine letters duiden op gehele of reële waarden; de hoofdletters duiden op een beperkt aantal opties.

symbool	parameter
n_m	maturiteitsdrempel
n_c	aantal kindknopen
A	splitsingsalgoritme
T_w	tijdsmeting
λ	wegingsfactor
T_m	modeltype
t_r	interval voor omgekeerd herstarten
F	terugkoppelingsmechanisme

beschreven kan deze tijdsmeting absoluut of relatief gebeuren. De "wegingsfactor" λ stelt de basis voor van de exponentiële wegingsfunctie $w(\delta_t) = \lambda^{\delta_t}$. Het "modeltype" T_m kan parametrisch zijn of niet-parametrisch. Voor een parametrisch model hebben we enkel de gaussiaanse verdeling gebruikt. Het niet-parametrische model gebeurt volgens de historische aanpak. Het "interval voor omgekeerd herstarten" t_r geeft de periode weer van volledig herstarten van het model waarbij de training omgekeerd verloopt in de tijd. Een waarde van ∞ wijst erop dat het model nooit herstart wordt. Tenslotte beschrijft het "terugkoppelingsmechanisme" F of er een artificiële achtste binaire trefprior z_8 gebruikt wordt.

6.5.3 Resultaten

Een globale optimalisatie van alle parameters voor iedere combinatie van $h \in \{1, 5, 25\}$ en $p_v \in \{0.01, 0.05\}$ en voor iedere combinatie van kandidaatpriors op basis van een exhaustieve zoektocht is niet haalbaar binnen een redelijke termijn. Daarom verloopt de parameterzoektocht in twee stappen. In een eerste stap leiden min of meer willekeurige experimenten tot een beperkt aantal waarden voor elke parameter die competitieve resultaten kunnen opleveren. In een tweede stap worden deze afzonderlijke waarden gecombineerd met elkaar tot een uitgebreide maar eindige verzameling die de basis vormen voor een exhaustieve zoektocht. Tabel 6.3 geeft een overzicht van al deze parametercombinaties.

De tabel is het resultaat van de grove proefondervindelijke optimalisatie in de eerste stap. Deze tabel laat al toe de volgende drie conclusies te nemen. Ten eerste heeft het genereren van meer dan 2 nieuwe knopen in de

Tabel 6.3: Een grove optimalisatie in een eerste stap leidt tot een aantal kandidaatwaarden voor een exhaustieve parameteroptimalisatie in een tweede stap.

symbool	waarden
n_m	100, 200, 300, 500, 1000
n_c	2
A	willekeurig, snel min-max, volledig min-max
T_w	relatieve tijdsmeting
λ	1, 0.9995, 0.999, 0.995, 0.99
T_m	gaussiaans, historisch
t_r	100, 200, 500, 1000, ∞
F	ja, nee

knoopgeneratie geen aanleiding tot een significante verbetering. Ten tweede doet relatieve tijdsmeting het altijd beter dan absolute tijdsmeting. Ten derde zijn de kandidaatgewichten relatief hoog in vergelijking tot de vervalfactor van RiskMetrics, waar respectievelijk $\lambda = 0.94$ en $\lambda = 0.97$ voor waarnemingen op dagelijkse en maandelijkse basis [114]. De optimalisatie van de parameters maakt gebruik van alle zeven kandidaatpriors.

Het interpreteren van de numerieke resultaten heeft als probleem dat de vijf numerieke criteria $\bar{m}$, m^+, χ^2, L en C gezamenlijk moeten geëvalueerd worden. Voor de eerste drie criteria kan eenvoudig een betrouwbaarheidsinterval berekend worden onder de hypothese dat het model het gedrag van de waarnemingen volledig kan onderscheppen. Het tweezijdig 92% betrouwbaarheidsinterval voor het waargenomen trefgemiddelde per venster $\bar{m}$ bedraagt $[0.783, 1.22]$ en $[4.52, 5.47]$ voor respectievelijk $p_v = 0.01$ en $p_v = 0.05$, waarbij $l = 100$ en $q = 60$ wordt gesteld. Voor het waargenomen maximum per venster is daarenboven $\Pr[m^+ \leq 5] = 96.84\%$ en is $\Pr[m^+ \leq 14] = 91.58\%$ voor respectievelijk $p_v = 0.01$ and $p_v = 0.05$. Tenslotte wordt het eenzijdig 95% betrouwbaarheidsinterval voor χ^2 gegeven door $[0, 124.34]$. De overige twee criteria, de verliesfunctie L en de kostfunctie C moeten allebei zo klein mogelijk zijn. Alle criteria beïnvloeden elkaar en ze moeten dan ook gezamenlijk geëvalueerd worden. Het invoeren van zoveel criteria is nodig om een fundamenteel zinvolle analyse van de value-at-risk mogelijk te maken.

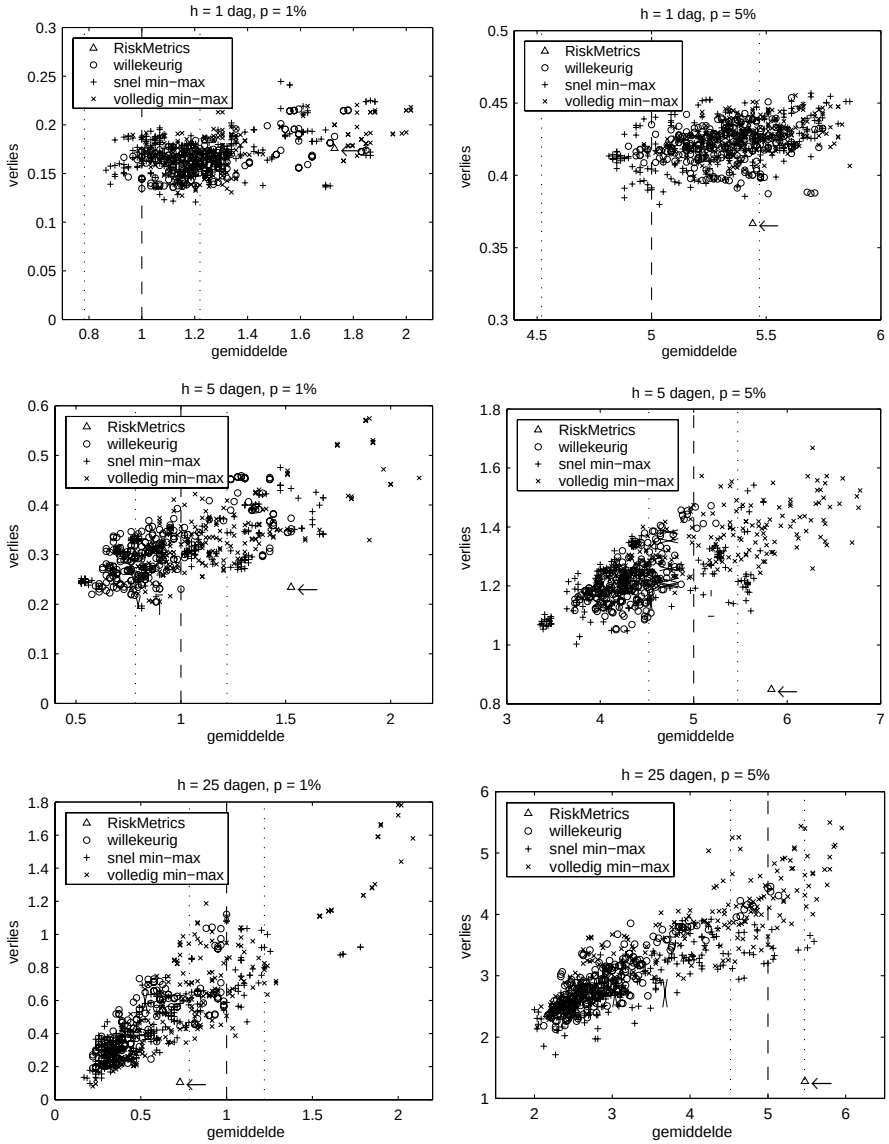
Het optimaliseren van een vrij groot aantal parameters op deze manier is heel gevoelig voor datagappen omdat het niet duidelijk is hoe robuust de optimale combinatie zal zijn voor het modelleren van de waarden van andere financiële instrumenten of andere perioden.

Optimalisatie van de parameters

Voor iedere combinatie van h en p_v geeft figuur 6.3 het verlies L samen met het gemiddelde $\bar{m}$ grafisch weer. Met elke parametercombinatie van tabel 6.3 stemt een resultaat in de grafiek overeen, met die uitzondering dat enkel de historische aanpak voorgesteld wordt. Voor elk splitsingsalgoritme wordt een ander symbool gebruikt. Het gesimuleerd resultaat volgens de aanpak van RiskMetrics wordt eveneens weergegeven op elke grafiek.

Voor elk van de zes gevallen is er een parametercombinatie beschikbaar die aanvaardbare resultaten behaalt, dit laatste betekent een waarde voor $\bar{m}$ dicht bij de verwachtingswaarde en een vrij lage waarde voor L. Gemiddeld zijn de resultaten voor een value-at-risk met probabiliteit $p_v = 0.01$ beter dan die voor $p_v = 0.05$, wat betekent dat historische modellering beter is om het probleem van de dikke staarten aan te pakken. De resultaten zijn ook beter naarmate de horizon h kleiner wordt. Dit kan verklaard worden door het feit dat de meeste priors de dynamiek op korte termijn beschrijven en dat er vrij weinig informatie in de contextruimte beschikbaar is over de dynamiek op lange termijn. Een vergelijking tussen de aanpak gebaseerd op contextmodellering en de aanpak volgens RiskMetrics leert dat het contextmodel het aanzienlijk beter doet voor de gevallen ($h = 1$, $p_v = 0.01$) en ($h = 5$, $p_v = 0.01$), en dat het contextmodel het aanzienlijk slechter doet voor het geval ($h = 25$, $p_v = 0.05$). Realiseer echter dat dit onderzoek als doel heeft om het gedrag van onwaarschijnlijke gebeurtenissen op korte termijn te modelleren. Het is eerder verrassend dat het snelle min-max algoritme voor knoopgeneratie het altijd optimaal of bijna optimaal doet. Slechts in enkele situaties haalt het volledige min-max algoritme lichtjes betere resultaten. Vermoedelijk heeft het volledige min-max algoritme meer potentieel, maar past het zich trager aan aan de data omdat niet alleen de buitengewone maar ook de gewone waarnemingen in rekening gebracht worden. Vooral voor grotere waarden van de horizon h worden de verschillen tussen de splitsingsalgoritmen groter.

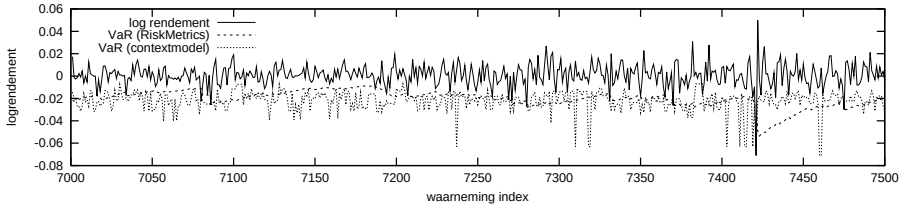
Tabel 6.4 geeft de numerieke resultaten weer op basis van de optimale parametercombinatie voor elk van de zes gevallen. De beste waarde voor het waargenomen gemiddelde $\bar{m}$ wordt in vet afgedrukt. Indien geen contextmodellering wordt toegepast, nemen de gewichten de standaardwaarden volgens RiskMetrics aan, dit betekent $\lambda = 0.94$ indien $h = 1$, $\lambda = 0.95$ indien $h = 5$ en $\lambda = 0.97$ indien $h = 25$. Als voor elke combinatie van h en p_v de optimale parametercombinatie gebruikt wordt, dan haalt historische contextmodellering consequent het beste resultaat met betrekking tot het gemiddelde $\bar{m}$. Maar het maximum m^+ en de χ^2-statistiek zijn meestal een stuk hoger. De beste verbetering wordt bekomen voor een korte horizon en een lage probabiliteit. Dit is omdat de indicatoren vooral gedrag op korte termijn beschrijven en omdat



Figuur 6.3: Voor elke combinatie van $h \in \{1, 5, 25\}$ en $p_v \in \{0.01, 0.05\}$ geeft de grafiek het verlies $L(\times 10^{-3})$ weer versus het waargenomen gemiddelde $\bar{m}$ en dit voor elke parametercombinatie (enkel historische modellering). De stippellijn toont het 92% betrouwbaarheidsinterval voor $\bar{m}$. Het resultaat volgens RiskMetrics is gemarkeerd als "$\Delta \leftarrow$".

Tabel 6.4: Numerieke resultaten voor de S&P500 (zonder de crash van 1987) op basis van de optimale parameters. De modellering is gaussiaans of historisch al of niet in combinatie met contextmodellering. Voor iedere combinatie van h en p_v is het beste resultaat voor $\bar{m}$ gemarkeerd in vet.

contexten	modeltype	$\bar{m}$	m^+	χ^2	$L(\times 10^{-3})$	C
I - Horizon $h = 1$, probabiliteit $p_v = 0.01$						
nee	gaussiaans	1.73	5	52.1	0.18	1.96
nee	historisch	2.59	5	222.6	0.23	1.93
ja	gaussiaans	1.53	6	95.81	0.20	1.97
ja	historisch	**1.08**	6	42.82	0.12	2.37
II - Horizon $h = 1$, probabiliteit $p_v = 0.05$						
nee	gaussiaans	5.44	9	20.9	0.37	1.41
nee	historisch	6.37	10	40.5	0.38	1.40
ja	gaussiaans	4.64	13	83.85	0.38	1.43
ja	historisch	**5.03**	14	115.84	0.38	1.42
III - Horizon $h = 5$, probabiliteit $p_v = 0.01$						
nee	gaussiaans	1.53	5	54.7	0.23	4.34
nee	historisch	2.80	6	384.0	0.44	4.10
ja	gaussiaans	1.44	13	571×10^6	0.40	4.78
ja	historisch	**1.00**	8	2.32×10^3	0.23	5.62
IV - Horizon $h = 5$, probabiliteit $p_v = 0.05$						
nee	gaussiaans	5.83	13	31.9	0.85	3.11
nee	historisch	6.51	11	57.9	0.92	3.09
ja	gaussiaans	5.71	21	1.04×10^6	1.40	3.29
ja	historisch	**4.92**	16	956.02	1.13	3.52
V - Horizon $h = 25$, probabiliteit $p_v = 0.01$						
nee	gaussiaans	0.73	5	55.9	0.11	8.03
nee	historisch	2.83	9	393×10^3	0.45	7.32
ja	gaussiaans	1.22	13	2.29×10^9	0.78	10.14
ja	historisch	**0.98**	12	39.3×10^6	0.54	11.17
VI - Horizon $h = 25$, probabiliteit $p_v = 0.05$						
nee	gaussiaans	5.47	17	22.0×10^3	1.28	5.74
nee	historisch	7.15	19	47.8×10^3	1.34	5.64
ja	gaussiaans	5.02	26	7.42×10^9	3.92	7.18
ja	historisch	**5.00**	19	261×10^3	3.51	7.46



Figuur 6.4: Een typisch verloop van het logrendement, samen met de value-at-risk volgens RiskMetrics (traag variërende streeplijn) en de value-at-risk volgens het voorgestelde contextmodel (grillige stippellijn) voor een horizon $h = 1$ en een probabiliteit $p_v = 0.01$. Let op het verschil tussen beide voorspellingen, vooral kort voor en na de waarneming met index 7421 (de crash van oktober 1987).

de niet-parametrische aanpak zich uitstekend leent voor het probleem van de dikke staarten. Merk op dat voor ($h = 1$, $p_v = 0.01$) noch de historische modellering zonder contexten noch de parametrische modellering met contexten erin slagen de resultaten van RiskMetrics te verbeteren, maar dat de werkelijke verbetering precies in de combinatie van het contextmodel met de historische aanpak ligt.

Figuur 6.4 illustreert het typisch gedrag van het logrendement en twee schattingen voor de value-at-risk voor de waarnemingen met index 7000 tot 7500 voor ($h = 1$, $p_v = 0.01$) op basis van de optimale parametercombinatie volgens tabel 6.4. Het verwachte aantal treffers over deze volatiele periode bedraagt precies 5. De grafiek toont een heel groot kwalitatief verschil tussen beide voorspellingen voor de value-at-risk. De value-at-risk volgens RiskMetrics behaalt 17 treffers, een onredelijk hoog aantal. Deze voorspelling wordt gekarakteriseerd door een trage terugval in periodes van lage volatiliteit, door het consequent verkeerd inschatten van extreem negatieve rendementen en door een overdreven toename onmiddellijk na een buitengewone waarneming. De value-at-risk volgens het contextmodel behaalt 8 treffers, wat aanvaardbaar is. Deze voorspelling slaagt er beter in de niet-stationariteiten te onderscheppen met betrekking tot buitengewoon gedrag, maar vertoont een bijzonder irregulier patroon wat veroorzaakt wordt door het voortdurend wisselen van contextklasse. Dit gedrag druist in tegen de intuïtief aanvaardbare notie van een traag variërend risico en zou kunnen geïnterpreteerd worden als een teken van slechte modellering. Maar het is een inherent gevolg van de gekozen modelvorming dat weliswaar gematigd zou kunnen worden indien meer gegevens beschikbaar zijn voor training en indien een gemengd model zou gebruikt worden.

De waarnemingen van figuur 6.4 bevatten eveneens de crash van 1987 ter

Tabel 6.5: Optimale parameters en hun gevoeligheid voor ($h = 1$, $p_v = 0.01$).

parameter	waarde	gevoeligheid
n_m	200	$+$
n_c	2	$-$
A	snel min-max	$+$
T_w	relatief	$+$
λ	0.99	$+$
T_m	historisch	$+$
t_r	200	$-$
F	nee	$-$

hoogte van index 7421. Merk op dat een venster rond deze waarnemingen weliswaar niet gebruikt wordt voor parameteroptimalisatie of training, maar wel ter validatie van het contextmodel. De aanpak volgens RiskMetrics slaagt er absoluut niet in de crash te voorspellen: de value-at-risk neemt alleen maar af in de periode voorafgaand aan de crash en schiet omhoog de dag na de crash. Het contextmodel daarentegen voorspelt meer en meer een hoge value-at-risk naarmate de dag van de crash nadert. Onmiddellijk na de crash wordt opnieuw een lage value-at-risk voorspeld wat aangeeft dat het gevaar op een groot risico voorbij is. Dit wijst erop dat het contextmodel aanvoelt dat er een periode van buitengewoon risico aankomt en dat het ook aanvoelt wanneer die periode weer voorbij is. Het valt wel op te merken dat het contextmodel op de dag van de crash een vrije lage value-at-risk voorspelt.

Tabel 6.5 vat de optimale parameters en de respectieve gevoeligheid voor deze parameters samen voor het geval ($h = 1$, $p_v = 0.01$). Hieruit kan afgeleid worden dat omwille van de contextmodellering relatief hoge gewichten kunnen gebruikt worden in vergelijking tot de gewichten van RiskMetrics. Bovendien behaalt de snelle min-max knoopgeneratie de beste resultaten, al is het verschil met de willekeurige knoopgeneratie en de volledige min-max knoopgeneratie beperkt. Het regelmatig herstarten van het contextmodel lijkt vruchten af te werpen maar het terugkoppelingsmechanisme leidt niet tot het verhoopte resultaat. De oorzaak van dit falen ligt vermoedelijk in het zuiver binaire karakter van deze conditionerende grootheid.

Rol van de priors

Het optimaliseren van de parameters is gebaseerd op alle vooraf beschreven priors. Maar niet alle priors zijn even belangrijk en het is dan ook leerrijk na te

Tabel 6.6: Optimale keuze voor de priors indien het aantal priors n_p beperkt is.

n_p	z_1	z_2	z_3	z_4	z_5	z_6	z_7	$\bar{m}$	$L(\times 10^{-3})$
0	-	-	-	-	-	-	-	1.08	0.16
1	-	-	-	X	-	-	-	1.98	0.16
2	-	-	-	-	-	X	X	1.00	0.12
3	-	-	X	-	X	X	-	1.00	0.14
4	-	X	-	-	X	X	X	1.02	0.14
5	X	X	-	-	X	X	X	1.00	0.14
6	X	X	X	X	X	-	X	1.07	0.15
7	X	X	X	X	X	X	X	1.08	0.13

gaan hoe goed elke mogelijke combinatie van zeven priors of minder het doet. Analyse van de optimale priorkeuze leert iets over de verborgen mechanismen die het marktrisico beïnvloeden.

Tabel 6.6 beschrijft de beste priorkeuze indien slechts een beperkt aantal priors n_p toegelaten is voor het geval ($h = 1$, $p_v = 0.01$). Hieruit blijkt dat het differentieel dividendrendement z_7, de differentiële falingsmarge in de interestvoeten z_6, de differentiële termijnstructuur z_5 en het rendement op 5 dagen z_2 de grootste voorspellende kracht hebben. Maar het valt op te merken dat naarmate het aantal priors n_p toeneemt, niet steeds dezelfde priors verkozen worden. Dit wijst erop dat er veel mutuele informatie is tussen de priors, maar dat is moeilijk te kwantificeren en te analyseren.

Robuustheid van de parameters

Het voorgestelde contextmodel telt een behoorlijk aantal parameters en al deze parameters worden geoptimaliseerd op slechts één reeks waarnemingen. Dit leidt tot optimale resultaten maar de aanpak is heel gevoelig aan datagappen. Het is dan ook de vraag of parameterwaarden die geoptimaliseerd zijn op basis van het verleden ook goede waarden blijven voor de toekomst.

Om dit te analyseren hebben we een beperkt experiment uitgevoerd waarbij de parameters onafhankelijk geoptimaliseerd worden over twee gescheiden tijdsintervallen, respectievelijk de eerste 6000 waarnemingen en de laatste 2000 waarnemingen. Niettegenstaande een vergelijking van de financiële gegevens in deze perioden aantoont dat ze sterk niet-stationair zijn, leidt een optimalisatie toch tot nagenoeg dezelfde waarden en enkel het splitsingsalgoritme verschilt. Dit is een aanwijzing dat de procedure om de parameters te optimaliseren redelijk robuust is.

Tabel 6.7: Numerieke resultaten voor de S&P500 waarbij de crash van 1987 wel in rekening gebracht wordt voor het geval ($h = 1$, $p_v = 0.01$). De modellering is gaussiaans of historisch al of niet in combinatie met contextmodellering. De parameters zijn niet geoptimaliseerd maar voorzichtig gekozen op basis van voorgaande experimenten. Het beste resultaat voor $\bar{m}$ is gemarkeerd in vet.

contexten	modeltype	$\bar{m}$	m^+	χ^2	$L(\times 10^{-3})$	C
nee	gaussiaans	2.00	5	120.9	0.75	2.00
nee	historisch	2.62	5	231.4	0.76	2.13
ja	gaussiaans	2.03	9	24.3×10^3	0.92	1.97
ja	historisch	**1.43**	8	2.29×10^3	0.73	2.51

De crash van 1987

Tot hiertoe werd een periode van waarnemingen rond de crash van 1987 uitgesloten voor training. Tabel 6.7 geeft een aantal numerieke resultaten voor ($h = 1$, $p_v = 0.01$) indien deze volatiele periode wel in rekening gebracht wordt voor de training. De parameters voor het contextmodel zijn niet exhaustief geoptimaliseerd maar gebaseerd op voorgaande experimenten. Indien geen contextmodellering gebruikt wordt is $\lambda = 0.94$ overeenkomstig de standaard volgens RiskMetrics.

Een overgang van de klassieke aanpak volgens RiskMetrics naar het historisch contextmodel toont een verbetering aan wat betreft het gemiddelde $\bar{m}$ en de verliesfunctie L, maar het maximum m^+, de χ^2-statistiek en de kostfunctie C gaan erop achteruit. Het valt op te merken dat een aantal van deze numerieke maten sterk niet-lineair zijn en hun waarde wordt hoofdzakelijk bepaald door extreme waarden. Dit extreem gedrag is precies geconcentreerd in de periode rond de crash van 1987. Het waargenomen gemiddelde $\bar{m}$ gaat erop vooruit, maar heeft dan ook geen last van deze niet-lineariteiten.

6.6 Samenvatting en eigen bijdragen

Dit hoofdstuk toont aan hoe contextmodellen hun nut kunnen bewijzen in andere toepassingen dan datacompressie. Meer bepaald wordt er dieper ingegaan op de financiële modellering en de risicoanalyse met de voorspelling van de value-at-risk als concrete toepassing. De uiteindelijke doelstellingen van datacompressie en financiële modellering zijn gelijkaardig: beide proberen het waargenomen gedrag zo efficiënt mogelijk te beschrijven of te voorspellen. Maar er zijn ook fundamentele verschillen, met name een lager aantal beschik-

bare waarnemingen voor training, de niet-gehele aard van de gegevens, andere reversibiliteitsvoorwaarden en een hoger aantal kandidaatgrootheden voor de contextvorming.

De verschillen tussen de doelstellingen en de geassocieerde gegevens leiden tot een aantal aanpassingen in het contextmodel. Het partitioneren van de toestandsruimte tot een beperkt aantal discrete contextklassen gebeurt aan de hand van een algoritme voor boomgestructureerde vectorkwantisatie. Op basis van historische waarnemingen van de Amerikaanse S&P500-aandelenindex worden de parameters van het model exhaustief geoptimaliseerd en gevalideerd. Deze waarnemingen overspannen een periode van meer dan 30 jaar, maar de periode rond de crash van 1987 wordt uitgesloten voor de training van het model. Om de voorspelling van de value-at-risk te evalueren worden een aantal numerieke criteria en bijhorende betrouwbaarheidsintervallen opgesteld.

Experimentele resultaten tonen aan dat historische contextmodellering het aanzienlijk beter doet dan de conventionele aanpak volgens RiskMetrics indien zowel de horizon als de probabiliteit van de value-at-risk heel laag is. De precieze kracht van de aanpak situeert zich in de combinatie van het invoeren van contexten, de teruggekoppelde zelfpartitionering van de contextruimte en de niet-parametrische modellering. Dit wijst erop dat het voorgesteld model er goed in slaagt de niet-stationariteiten te onderscheppen en bovendien in staat is de geobserveerde dikke staarten in de verdeling van het rendement te modelleren. In tegenstelling tot de aanpak volgens RiskMetrics slaagt het contextmodel er behoorlijk in om de crash van 1987 te anticiperen en vervolgens snel te recupereren na de crash.

Het historisch contextmodel behoort tot de klasse van blinde modellen omdat er weinig of geen veronderstellingen gemaakt worden omtrent het onderliggend gedrag. Het volledige gedrag wordt uitsluitend uit de waarnemingen afgeleid door interne training op basis van waarnemingen in het verleden. Deze aanpak leert dat de fundamentele factoren zoals het differentieel dividendrendement, de differentiële falingsmarge in de interestvoeten en de differentiële termijnstructuur een grotere voorspellende kracht hebben dan de technische factoren zoals het rendement op korte en middellange termijn.

De voorgestelde aanpak mag niet als een volledig alleenstaande methode beschouwd worden. In een reële toepassing is het aangewezen om de voorspellingen volgens meerdere modellen te combineren en aan te vullen met verworven kennis van de marktmechanismen.

Naast dit onderzoek naar entropiegestuurde risicoanalyse hebben we ook enig onderzoek verricht naar de optimalisatie van het portefeuillebeheer op basis van entropie [42, 71]. Deze aanpak laat toe een verfijnde strategie op te

bouwen, maar leidt niet tot hogere winst of verlies. Deze resultaten komen niet aan bod in dit werk.

Dit onderzoek gebeurde in samenwerking met de vakgroep financiële economie van de Universiteit Gent en heeft geleid tot één bijdrage op een internationale conferentie [249]. Verder onderzoek heeft vervolgens geleid tot een publicatie in een internationaal tijdschrift [55]. Het programmeerwerk in het kader van dit onderzoek en het uitvoeren van de experimenten werd uitgevoerd door collega Steven Van Assche.

Hoofdstuk 7

Besluit

Dit proefschrift handelt over de verliesloze compressie van beelden en probeert inzicht te verschaffen hoe de onderliggende modelvorming kan geoptimaliseerd en uitgebreid worden. In dit hoofdstuk vatten we de belangrijkste conclusies van het onderzoek samen en geven we een overzicht van de originele bijdragen.

Hoofdstuk 2 vormt een inleidend hoofdstuk dat beeldcompressie van buitenaf benadert. Het algemeen compressieschema beschrijft hoe nagenoeg alle moderne technieken samengesteld zijn uit een blok dat de beeldgegevens modelleert en een blok dat de pixels codeert op basis van de resultaten van het eerste blok. De grootheid entropie drukt de onvoorspelbaarheid van de waarnemingen uit, dit in tegenstelling tot de redundantie die de voorspelbaarheid uitdrukt. De entropie vormt een ondergrens voor de verwachte codelengte per symbool en wordt om die reden als een maat voor de informatie aanzien. De natuurlijke band met de informatietheorie en de complexiteitstheorie laten toe een aantal algemene uitspraken te doen die gelden voor elke compressietechniek.

Vervolgens gaan we in hoofdstuk 3 dieper in op de meest gebruikte codes en hun eigenschappen. Entropiecodes zijn uniek decodeerbaar en baseren de codelengte voor elke bronwaarde op de geassocieerde waarschijnlijkheid van optreden. Huffmancodes zijn optimaal onder de randvoorwaarde dat voor elke bronwaarde een codewoord van gehele lengte gedefinieerd is. De constructie ervan verloopt snel maar de efficiëntie schiet tekort indien één bronwaarde een heel hoge waarschijnlijkheid van optreden heeft of indien het bronalfabet een beperkte lengte heeft. Aritmetische codes kennen deze tekortkomingen niet omdat ze één codewoord voor de gehele bronsequentie definiëren. De constructie ervan verloopt een stuk trager, maar ze hebben de elegante eigenschap dat hun verwachte codelengte de entropie arbitrair dicht kan benaderen.

Daarnaast bestaan nog een aantal geparametriseerde codes die een compromis vormen tussen snelheid en efficiëntie.

De state of the art van verliesloze compressie evenals een aantal optimalisaties voor beelden in de drukvoorbereidingsindustrie worden beschreven in hoofdstuk 4. Veel technieken voor de verliesloze compressie van beelden zijn geworteld in het domein van de tekstcompressie, al is het contextmodel zelf eerst ingevoerd voor binaire beelden. De beste technieken voor teksten zijn gebaseerd op het mengen van contextmodellen van variabele diepte. Als originele bijdrage stellen we een nieuw paradigma voor voor de verliesloze compressie van beelden dat een uitbreiding is van het algemeen compressieschema. Voor binaire beelden zijn er de JBIG-standaarden die contextmodellering combineren met multiresolutiedecompositie en patroonherkenning. Voor monochrome beelden zijn er enerzijds de sequentiële technieken zoals JPEG-LS en CALIC die gebaseerd zijn op niet-lineaire voorspelling en foutmodellering aan de hand van contexten. Anderzijds zijn er de multiresolutietechnieken zoals JPEG2000 die gebruik maken van wavelets en waarbij de nadruk meer ligt op schaalbaarheid, robuustheid en flexibiliteit dan op efficiëntie. De beschreven technieken doen het lang niet altijd optimaal voor beelden uit de drukvoorbereidingsindsutrie en om die reden stellen we een drietal eenvoudige optimalisaties voor. Het gebruik van een adaptieve referentielijn voor Groep 4 levert een winst van circa 40% op voor halftoonbeelden. Statische lineaire tonale decorrelatie van CMYK-beelden levert een winst op van ongeveer 10%, wat in de buurt komt van een aantal geavanceerde technieken. De redundante voorstelling van de CMYK-kleurenruimte laat toe een nieuw criterium te definiëren voor bijna-verliesloze compressie. Elk van deze drie optimalisaties vormt een eigen bijdrage. We dienen hierbij op te merken dat er een inspanning geleverd is om de state of the art van beeldcompressie up-to-date te houden. Er worden dan ook een aantal technieken in besproken die chronologisch pas verschenen zijn na de publicatie van de eigen bijdragen, die het onderwerp uitmaken van de volgende hoofdstukken.

Zo beschrijft hoofdstuk 5 hoe het contextmodel nog verregaand kan geoptimaliseerd worden voor halftoonbeelden. Een geoptimaliseerd adaptief contextsjabloon heeft tot doel de statistische gevolgen van het halftoonproces te onderscheppen. Een gretig zoekproces leidt tot een nagenoeg optimaal sjabloon maar is niet haalbaar voor praktische doeleinden. Een sjabloon op basis van een schatting van de autocorrelatie biedt een uniek compromis. Voor klassieke halftoonbeelden zijn resultaten op basis van het autocorrelatiegebaseerd sjabloon 20–50% beter dan voor JBIG1 en slechts 4–8% minder goed dan het nagenoeg optimale sjabloon. Een optimale implementatie in software toont aan dat de constructie van het autocorrelatiegebaseerd sjabloon in een tijdsduur

van grootteorde één seconde kan verlopen. Een geoptimaliseerde implementatie in software toont aan dat het adaptief maken van het sjabloon niet tot een wezenlijke vertraging van het codeerproces moet leiden. De beperking van het aantal adaptieve sjabloonpixels in de JBIG2-standaard om snelheidsredenen is dan ook overbodig. Maar ook voor een efficiënte implementatie in hardware vormt de adaptiviteit van het sjabloon geen obstakel. Op een 700 MHz platform worden in software snelheden gehaald van ongeveer 14×10^6 en 12×10^6 pixels per seconde voor respectievelijk het coderen en het decoderen. Op een 400 MHz hardware-architectuur bedragen deze snelheden respectievelijk 340×10^6 en 170×10^6 pixels per seconde. De efficiëntie van de methode kan nog verder opgedreven worden door gebruik te maken van een onvolledige gebalanceerde tweeboom als contextmodel. De ontwerpvrijheid in het sjabloon laat bovendien toe dezelfde techniek te gebruiken voor de residucodering van contone beelden, wat aanleiding geeft tot een vrij universele aanpak voor verliesloze beeldcompressie. De originele bijdragen zijn het autocorrelatiegebaseerd sjabloon, de snelle software- en hardware-implementatie, het boommodel met adaptief contextsjabloon en de uitbreiding naar universele sjabloongebaseerde residucodering. Het gedetailleerde ontwerp van de hardware-implementatie werd uitgevoerd door collega's binnen de onderzoeksgroep.

Dat de modelvorming op basis van contexten ook benut kan worden in andere wetenschapsdomeinen zoals de financiële risico-analyse toont hoofdstuk 6 aan. De value-at-risk is een maat voor het geschatte verlies van een financieel instrument over een voorgedefinieerde horizon en tegen een lage vooropgestelde probabiliteit. We stellen een aangepast contextmodel voor dat toelaat een voorspelling te maken zonder er kennis van buitenaf in te stoppen. De aanpassingen situeren zich in het opgeven van de causaliteit, het gebruik van externe priors, de zelforganiserende natuur van de boomgestructureerde vectorkwantisatie, de gewijzigde groei- en trainingsparameters, het gebruik van parametrische modellering naast historische modellering en een aantal criteria voor de evaluatie van de voorspelling van de value-at-risk. Experimentele resultaten op historische gegevens tonen een grillig verlopende maar efficiënte voorspelling van de value-at-risk. De techniek onderscheidt zich van de aanpak volgens RiskMetrics door een groter anticiperend karakter in de periode vóór een extreme gebeurtenis en door een snellere herstelling na deze gebeurtenis. De gehele aanpak en alle wijzigingen nodig voor de risico-analyse zijn originele bijdragen. De implementatie van het model werd uitgevoerd door een collega binnen de onderzoeksgroep.

Samengevat heeft het onderzoek voorgesteld in dit werk geleid tot de volgende originele bijdragen: een uitgebreid overzicht van de state of the art van verliesloze beeldcompressie vergezeld van een taxonomie en een vernieuwd

paradigma, enkele directe uitbreidingen van deze technieken voor beelden uit de drukvoorbereidingsindustrie, een geavanceerde compressietechniek voor halftoonbeelden met semi-adaptieve sjabloongeneratie, een snelle implementatie hiervan in zowel software als hardware die aantoont dat de adaptiviteit niet ten koste moet gaan van de snelheid, een aantal uitbreidingen van deze techniek voor niet-binaire beelden en het uitbreiden van contextmodellering voor andere doeleinden zoals de voorspelling van de value-at-risk voor economische risicoanalyse.

Bijlage A

Testbeelden

De testbeelden in dit werk hebben sterk afwijkende eigenschappen en komen van diverse bronnen. Hierna volgt een beknopt overzicht van de gebruikte beelden en hun oorsprong.

De naamconventie “<scène>-<kleur>{-<detail>}” wordt hierbij consistent gehanteerd. De beelden zijn georganiseerd volgens hun kleurrepresentatie “<kleur>”, die de waarden kan aannemen uit tabel A.1. Ter referentie worden een aantal scènes “<scène>” verkleind weergegeven in deze bijlage. Afwijkende en discriminerende eigenschappen van een beeld worden weergegeven in “<detail>”. De afmetingen van een beeld zijn weergegeven volgens de conventie “<breedte> × <hoogte>”. De notatie “*” op het einde van naam duidt op een uitsnijding van het beeld. Enkele van de meest gebruikte beelden krijgen een alias toegewezen.

Tabel A.1: Overzicht van de gebruikte kleurenruimtes en hun afkortingen.

<kleur>	bitdiepte	kleurenruimte
b	1	binair (geen halftoonbeeld)
h	1	halftoonbeeld
l	8, 10, 12	luminantie (grijswaardenbeeld)
r, g, b	8	rood, groen, blauw per component
c, m, y, k	8	cyaan, magenta, geel, zwart per component
lab	24	CIEL*a*b*
yuv	24	YUV
rgb	24	RGB
cmyk	32	CMYK

A.1 Tekstbestanden

Om de efficiëntie van algemene verliesloze compressietechieken te evalueren wordt gebruik gemaakt van een tekstcorpus. Enkele corpora zijn in de literatuur voorgesteld en als ad hoc standaard geadopteerd. Het “Calgary corpus” bevat 14 bestanden van verschillende formaten: ongeformatteerde Engelse tekst, geformatteerde Engelse tekst, broncode, objectcode, binaire data en een binair beeld [14]. Na verloop van tijd werden nieuwe technieken meer en meer geoptimaliseerd voor dit corpus. Bovendien was de relatieve verhouding van de samenstelling en de lijst van gebruikte formaten niet langer representatief. Dit resulteerde in het “Canterbury corpus”, dat 11 bestanden bevat [5]. De samenstelling is gelijkaardig, maar bevat nu ook formaten als Microsoft Excel en HTML. Tabel A.2 schetst een overzicht van de testbestanden en hun eigenschappen.

Tabel A.2: Overzicht van de bestanden van het Calgary en het Canterbury corpus.

Calgary			Canterbury		
bestand	formaat	grootte	bestand	formaat	grootte
bib	Unix refer	111 261	alice29.txt	tekst	152 089
book1	tekst	768 771	asyoulik.txt	tekst	125 179
book2	Unix troff	610 856	cp.html	HTML	24 603
geo	numeriek	102 400	fields.c	C	11 150
news	Usenet	377 109	grammar.lsp	LISP	3 721
obj1	VAX exe	21 504	kennedy.xls	Excel	1 029 744
obj2	Apple exe	246 814	lcet10.txt	tekst	426 754
paper1	Unix troff	53 161	plrabn12.txt	tekst	481 861
paper2	Unix troff	82 199	ptt5	fax	513 216
pic	beeld	513 216	sum	Sparc exe	38 240
progc	C	39 611	xargs.1	Unix man	4 227
progl	LISP	71 646			
progp	Pascal	49 379			
trans	transcript	93 695			

A.2 Binaire beelden

De binaire beelden stellen ofwel tekstuele informatie voor, ofwel gerasterde componenten van kleurenbeelden. Een niet gebruikt type is dat van binaire

beelden die gerasterde tekst aan hoge resolutie voorstellen. Voor verliesloze compressie is het ontstaansproces van belang omdat dit een grote impact heeft op de hoeveelheid ruis. We onderscheiden de door de CCITT gestandaardiseerde tekstbeelden, de JBIG Stockholm testbeelden en verder halftoonbeelden afkomstig van de JBIG Stockholm testset, de BG testbeelden[1] en geproduceerd aan de hand van ADOBE PHOTOSHOP.

A.2.1 CCITT tekstbeelden

De sets van tekstbeelden zijn samengesteld bij het ontwikkelen van standaarden voor faxcompressie (CCITT T.4 en T.6, JBIG en JBIG2). De beelden stellen typische zakendocumenten en technische documenten voor aan een beperkte resolutie. Alle 8 CCITT-testbeelden hebben dezelfde afmetingen en resolutie. Ze bevatten elk 1728×2376 pixels aan een resolutie van 200 ppi. Figuur A.1 geeft deze beelden weer aan sterk verlaagde resolutie. Alle beelden zijn ontstaan door reële documenten in te scannen. Deze beelden zijn gelijkaardig maar niet volkomen identiek aan de officiële NCS testbeelden die 1728×2339 pixels bevatten aan 200 ppi, en ook bestaan aan een resolutie van 300, 400 en 600 ppi[2]. In de literatuur heerst er verwarring over de precieze afkomst en resolutie van de testbeelden.

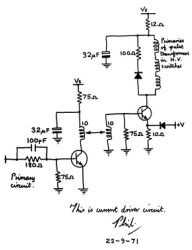
A.2.2 JBIG Stockholm testbeelden

Voor het opstellen van de JBIG-standaard diende een set van binaire testbeelden van diverse oorsprong als vergelijkingspunt [6]. Alle beelden hebben een resolutie van 400 dpi. Dit is een halvering van hun oorspronkelijke resolutie, maar helaas zijn de oorspronkelijke beelden niet beschikbaar. De set kan opgesplitst worden in categorieën van verschillende aard. Het gaat om tekstbeelden (“s0{1,2,3,5}a-b”), digitale halftoonbeelden (“s04{a–d}-b” en “s09a-b”), een gemengd beeld bestaande uit gescande tekst met halftoongebieden (“s06a-b”), gedigitaliseerd lijnwerk (“s07a-b”) en digitaal gegenereerd lijnwerk (“s{08,10}a-b”). De halftoonbeelden van deze set komen in volgende paragraaf meer gedetailleerd aan bod.

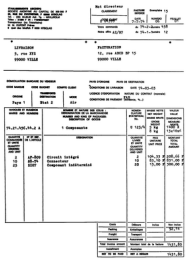
[1] Deze beelden zijn afkomstig van de firma Barco Graphics NV, die momenteel de naam Esko-Graphics NV draagt.

[2] Copyright National Communications Systems, te verkrijgen als ITU-T Recommendation T.24 Encl. (11/94).

ccitt1-b

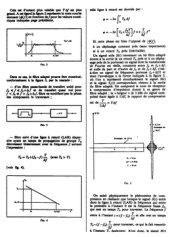


ccitt2-b

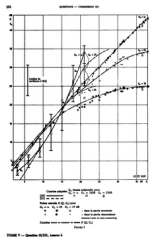


ccitt3-b

ccitt4-b



ccitt5-b



ccitt6-b

ccitt7-b

ccitt8-b

Figuur A.1: De CCITT testbeelden stellen zakendocumenten en tekstdocumenten voor.

A.2.3 Halftoonbeelden

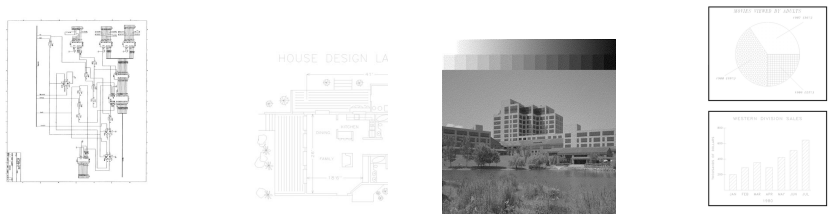
Halftoonbeelden ontstaan bij het rasteren van de kleurcomponenten. Rasteren komt voor in twee vormen: (i) klassiek rasteren of AM-rasteren, waarbij het rasterproces dots van veranderlijke grootte op een geroteerd regelmatig rooster plaatst, en (ii) stochastisch rasteren of FM-rasteren, waarbij het rasterproces dots van constante grootte op variabele posities plaatst. De rasterparameters zijn sterk uiteenlopend.

JBIG Stockholm halftoonbeelden

De JBIG Stockholm testset bevat in totaal vijf halftoonbeelden, namelijk “s04{a–d}-b” en “s09a-b”. De eerste vier zijn klassieke halftoonbeelden die dezelfde scène voorstellen maar ze hebben wisselende halftoonparameters (dotvorm en dotgrootte). Het vijfde beeld is een klein stochastisch halftoonbeeld. De halftoonbeelden hebben een relatief lage resolutie en zijn niet representatief voor de hedendaagse drukvoorbereidingsindustrie. Ze komen hier aan bod omdat er meerdere onderzoeksresultaten voor gepubliceerd zijn. Fi-

s01a-b 3072 × 4352 | s02a-b 3072 × 4352 | s03a-b 3072 × 4352 | s05a-b 3072 × 4352 | s06a-b 3072 × 4352

s04a-b 3072 × 2048 | s04b-b 3072 × 2048 | s04c-b 3072 × 2048 | s04d-b 3072 × 2048



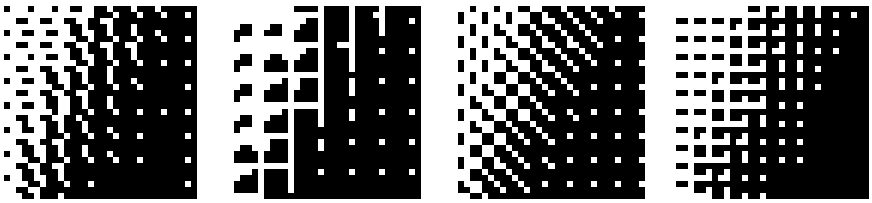
s07a-b 3072 × 4352 | s08a-b 3072 × 3040 | s09a-b 1024 × 1024 | s10a-b 4096 × 5856

Figuur A.2: De JBIG Stockholm testbeelden. De schalingsfactor van deze weergave is niet constant over alle beelden.

guur A.3 geeft een sterke uitvergroting van telkens dezelfde uitsnijding van elk van deze klassieke halftoonbeelden.

BG halftoonbeelden

Een aantal halftoonbeelden ter onze beschikking gesteld door de firma Barco Graphics NV (nu Esko-Graphics NV), afgekort als “BG”. Elk van hen stelt de cyaancomponent voor van “musicians-cmyk”, gerasterd volgens een ander kwaliteitsraster. De component, kortweg genoteerd als “mc”, wordt weergegeven links in figuur A.4. Tabel A.3 geeft een overzicht van de testbeelden en hun halftoonparameters. Deze beelden komen heel vaak terug in de experimenten en zijn daarom voorzien van een alias. Omdat beeld “mc-h-680-ro” gerasterd



s04a-b* s04b-b* s04c-b* s04d-b*

Figuur A.3: Sterke uitvergroting van telkens dezelfde uitsnijding (32×32) voor de klassieke JBIG Stockholm halftoonbeelden met een resolutie van 400 dpi.

musicians-c musicians-m musicians-y musicians-k

Figuur A.4: De vier componenten cyaan (C), magenta (M), geel (Y) en zwart (K) van "musicians-cmyk".

is op 45°, vertoont het periodieke structuren die samenvallen met karaktergrenzen (8 bit). Om deze redenen is dit beeld weinig geschikt voor testdoeleinden. Figuur A.5 toont een uitvergroting van elk van de hogeresolutiebeelden.

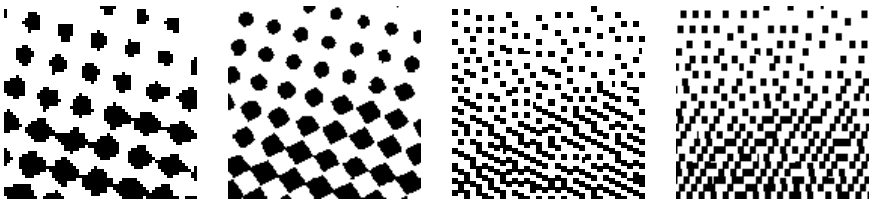
Elk van deze testbeelden is rechtstreeks digitaal gegenereerd op basis van de cyaancomponent "musicians-c". Om halftoonbeelden te simuleren die gegenereerd worden door een scanner, werd elk van de hogeresolutiebeelden onderworpen aan twee extra processtappen. Hierbij worden ze eerst effectief belicht op film en vervolgens opnieuw ingescand. Bij het inscannen wordt het testbeeld bewust onderworpen aan een minieme rotatie. De extensie "-sc" in de benaming verwijst naar deze extra processtappen.

Photoshop halftoonbeelden

De BG testbeelden zijn maar enkele typische voorbeelden van halftoonbeelden. Daarnaast hebben we nog een aantal halftoonbeelden gecreëerd op basis van de rasteralgoritmen die beschikbaar zijn in ADOBE PHOTOSHOP. De visuele kwaliteit van die rasters is lager dan van de BG rasters, maar voldoende om een aantal extra tests uit te voeren voor verliesloze compressie.

Tabel A.3: De BG halftoonbeelden en hun rasterparameters. De resolutie is uitgedrukt in dpi. Het rastertype beschrijft de dotvorm, de lijnfrequentie en de rasterhoek (klassiek) of diffusiealgoritme (stochastisch, bijvoorbeeld "Monet-1").

naam (alias)	grootte	resolutie	rastertype
I - Digitaal gegenereerd			
mc-h-680-ro (ro5k)	5054×4927	680	rond, 64 lpi, 45°
mc-h-1270-ro (ro10k)	9525×10795	1270	rond, 110 lpi, 75°
mc-h-2540-ro (ro20k)	19050×21590	2540	rond, 135 lpi, 75°
mc-h-1270-mo (mo10k)	9525×10795	1270	Monet-1, 40μ
mc-h-2540-mo (mo20k)	19050×21590	2540	Monet-1, 40μ
II - Na belichten op film en inscannen			
mc-h-680-ro-sc	5054×4921	680	rond, 64 lpi, 45°
mc-h-1270-ro-sc	9403×10670	1270	rond, 110 lpi, 75°
mc-h-2540-ro-sc	18804×21344	2540	rond, 135 lpi, 75°
mc-h-1270-mo-sc	9412×10682	1270	Monet-1, 40μ
mc-h-2540-mo-sc	18826×21366	2540	Monet-1, 40μ



mc-h-1270-ro* mc-h-2540-ro* mc-h-1270-mo* mc-h-2540-mo*

Figuur A.5: Sterke uitvergroting van telkens dezelfde uitsnijding (respectievelijk 64×64 pixels en 128×128 pixels) voor de BG halftoonbeelden van resolutie 1270 en 2540 dpi.

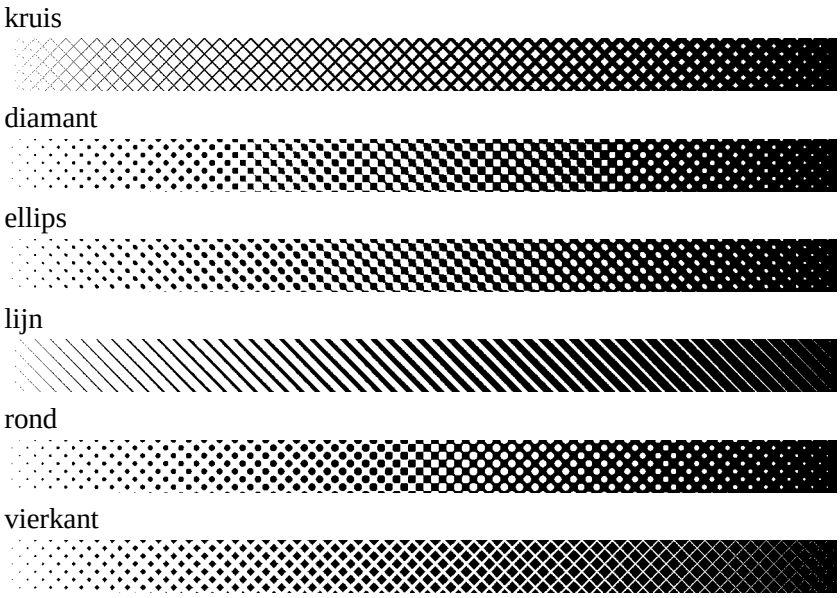
Tabel A.4 bevat de lijst van gegenereerde testbeelden en hun rasterparameters. De naamconventie is "mc-h-<resolutie>-<dottype>-<lijnfrequentie>-<rasterhoek>". Gebaseerd op het referentiebeeld "mc-h-1270-rd-110-15" wordt voor elk van de rasterparameters een aantal varianten voorgesteld. De beschikbare dottypes worden grafisch weergegeven in figuur A.6.

Tabel A.4: De ADOBE PHOTOSHOP halftoonbeelden en hun rasterparameters. Alle beelden zijn klassiek gerasterd. Het referentiebeeld hoort in elke variatieklasse terug thuis, maar wordt niet herhaald.

naam	grootte	resolutie	rastertype
Referentie			
mc-h-1270-rd-110-15	9413×10683	1270	rond, 110 lpi, 15°
I - Variatie volgens resolutie			
mc-h-0300-rd-075-15	2224×2524	300	rond, 75 lpi, 15°
mc-h-0600-rd-090-15	4447×5047	600	rond, 90 lpi, 15°
mc-h-0900-rd-100-15	6671×7571	900	rond, 100 lpi, 15°
mc-h-1200-rd-110-15	8894×10094	1200	rond, 110 lpi, 15°
mc-h-1500-rd-115-15	11118×12618	1500	rond, 115 lpi, 15°
mc-h-1800-rd-120-15	13342×15142	1800	rond, 120 lpi, 15°
mc-h-2400-rd-130-15	17789×20189	2400	rond, 130 lpi, 15°
mc-h-3000-rd-140-15	22236×25236	3000	rond, 140 lpi, 15°
mc-h-3600-rd-150-15	26683×30000	3600	rond, 150 lpi, 15°
II - Variatie volgens dottype			
mc-h-1270-cr-110-15	9413×10683	1270	kruis, 110 lpi, 15°
mc-h-1270-dm-110-15	9413×10683	1270	diamant, 110 lpi, 15°
mc-h-1270-el-110-15	9413×10683	1270	ellips, 110 lpi, 15°
mc-h-1270-ln-110-15	9413×10683	1270	lijn, 110 lpi, 15°
mc-h-1270-sq-110-15	9413×10683	1270	vierkant, 110 lpi, 15°
III - Variatie volgens rasterhoek			
mc-h-1270-rd-110-00	9413×10683	1270	rond, 110 lpi, 0°
mc-h-1270-rd-110-07	9413×10683	1270	rond, 110 lpi, 7.5°
mc-h-1270-rd-110-22	9413×10683	1270	rond, 110 lpi, 22.5°
mc-h-1270-rd-110-30	9413×10683	1270	rond, 110 lpi, 30°
mc-h-1270-rd-110-37	9413×10683	1270	rond, 110 lpi, 37.5°
mc-h-1270-rd-110-45	9413×10683	1270	rond, 110 lpi, 45°

A.3 Monochrome beelden

De monochrome beelden kunnen een luminantiecomponent voorstellen, een chrominantiecomponent of een kleurcomponent van een meercomponentenbeeld. We onderscheiden de testbeelden afkomstig van JPEG, medische testbeelden en kleurcomponenten.



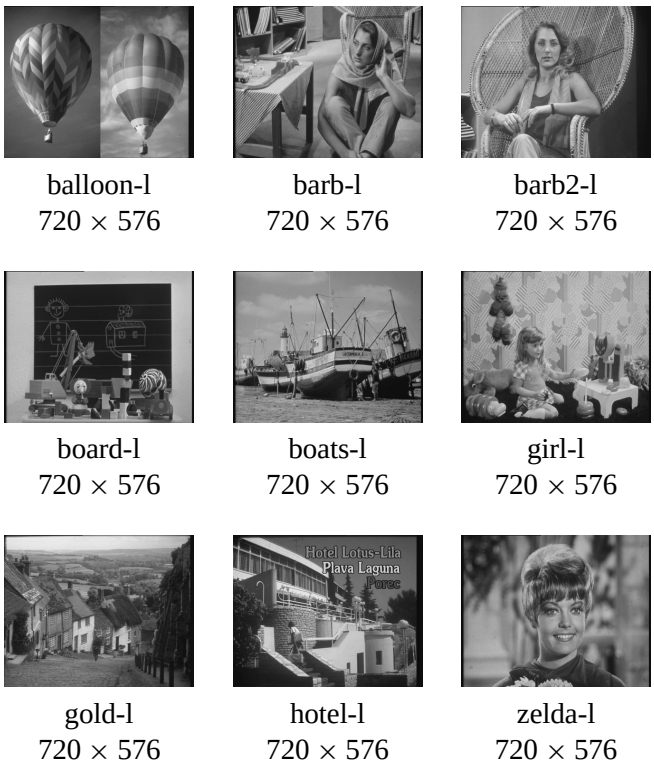
Figuur A.6: Een voorbeeld van een *helling* (*E.* ramp) voor elk van de dottypes van de beelden gerasterd aan de hand van ADOBE PHOTOSHOP.

A.3.1 JPEG testbeelden

Bij het tot stand komen van de JPEG-standaard werd een aantal beelden voorgesteld [119]. Ze stellen allemaal een luminantiecomponent (Y) voor. De afmetingen van elk van de beelden is 720×576 met een bitdiepte van 8 bit. Figuur A.7 geeft elk van deze beelden grafisch weer.

A.3.2 Medische beelden

Medische beelden onderscheiden zich van traditionele gescande beelden door hun ontstaansproces en dynamisch bereik. Veelal volgen de beelden uit een complex reconstructieproces en daarom worden ze vaak tomografische beelden genoemd. Bovendien is het dynamisch bereik van de beelden dikwijls veel groter, gaande van 8 bit, over 10 en 12 bit tot 16 bit. Figuur A.8 geeft een aantal testbeelden weer evenals hun afmetingen. De beginletters van elk beeld verwijzen naar de gebruikte beeldvormingsmodaliteit. Al zijn deze beelden typisch voorbeelden voor medische beeldvorming, de testset is te klein om representatief te zijn voor de typische ziekenhuisomgeving. Er komen geen datasets in voor die een 3D-volume voorstellen. Het onderzoek uitgevoerd in



balloon-l
720 × 576

barb-l
720 × 576

barb2-l
720 × 576

board-l
720 × 576

boats-l
720 × 576

girl-l
720 × 576

gold-l
720 × 576

hotel-l
720 × 576

zelda-l
720 × 576

Figuur A.7: Elk van de JPEG testbeelden stelt een luminantiecomponent (Y) voor.

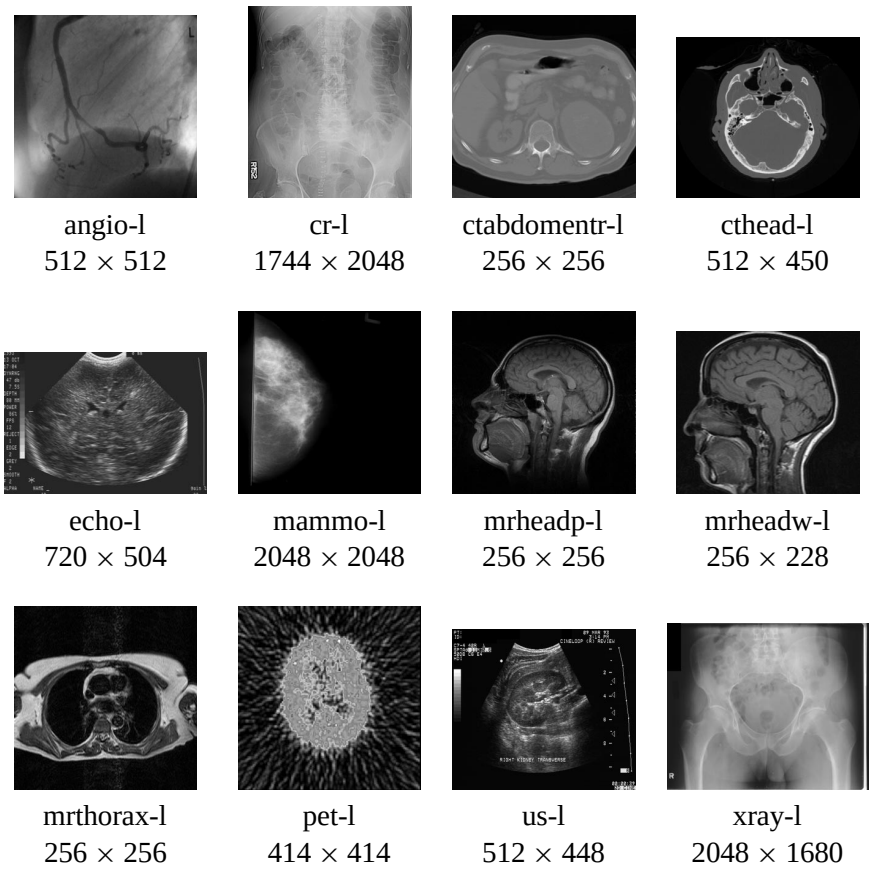
het kader van dit proefschrift omvat aanzienlijk meer testbeelden, onder andere ook beelden met een grotere bitdiepte en 3D-volumes [61, 65, 256].

A.3.3 Kleurcomponenten

De kleurcomponenten zijn gebaseerd op de individuele componenten van de meercomponentenbeelden. Figuur A.4 toont de componenten van “musicians-cmyk”. Voor de andere beelden geven we de componenten niet expliciet weer.

A.4 Meercomponentenbeelden

De meercomponentenbeelden stellen een scène voor in een kleurenruimte zoals RGB (rood, groen en blauw), CIEL*a*b*, YUV of CMYK (cyaan, magenta, geel en zwart). We onderscheiden de standaard RGB-beelden, de JPEG-LS testbeelden van ISO, de BG testbeelden en de recente SCID testbeelden van



Figuur A.8: Een aantal medische testbeelden.

ISO.

A.4.1 Standaard RGB-beelden

Een aantal klassiekers onder de RGB-testbeelden wordt weergegeven in figuur A.9. Alle beelden bevatten 512 × 512 pixels. Niet alleen hun kleurenweergave maar ook hun luminantie komt veel voor als testbeeld.

A.4.2 JPEG-LS testbeelden

Bij het ontwikkelen van JPEG-LS stelde ISO een aantal testbeelden ter beschikking van diverse oorsprong. De set bevat niet alleen grijswaarden- en kleurenbeelden van natuurlijke scènes, maar ook digitaal gegenereerde beel-

baboon-rgb
512 × 512

f16-rgb
512 × 512

lena-rgb
512 × 512

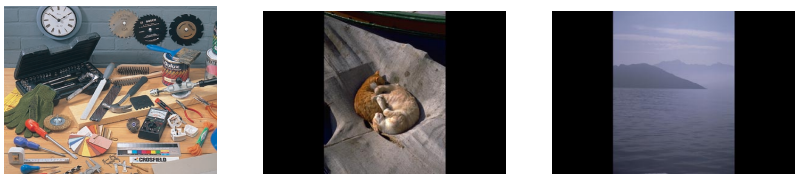
peppers-rgb
512 × 512

Figuur A.9: De klassieke RGB-testbeelden.

bike-cmyk
2048 × 2560

cafe-cmyk
2048 × 2560

woman-cmyk
2048 × 2560



tools-cmyk
1524 × 1200

cats-rgb
3072 × 2048

water-rgb
3072 × 2048

Figuur A.10: De contone testbeelden door ISO ter beschikking gesteld voor het ontwikkelen van JPEG-LS.

den, medische beelden en samengestelde beelden. Figuur A.10 geeft een aantal van de contone kleurenbeelden weer en beschrijft de afmetingen en de kleurenruimte. Het testbeeld “bike3-cmyk” stond niet tot onze beschikking en ontbreekt dan ook in deze set.

musicians-cmyk
1853 × 2103

scid0-cmyk
2048 × 2560

scid3-cmyk
2048 × 2560

koekjes1-cmyk
827 × 591

koekjes2-cmyk
839 × 638

timep-cmyk
339 × 432

fogra-cmyk
780 × 1002

Figuur A.11: De BG CMYK-testbeelden.

A.4.3 BG testbeelden

De firma Barco Graphics NV (nu Esko-Graphics NV) stelde ons eveneens een aantal CMYK testbeelden ter beschikking (afgekort als “BG”). Een aantal beelden, zoals bijvoorbeeld “musicians-cmyk” zijn afkomstig van dezelfde analoge testset als de beelden uit de volgende paragraaf. Niettegenstaande ze dezelfde scène voorstellen, gaat het wel degelijk om andere digitale beelden. Figuur A.11 geeft de beelden en hun afmetingen weer.

A.4.4 ISO SCID testbeelden

ISO heeft een standaard uitgebracht die exclusief uit testbeelden bestaat [110]. Deze standaard bevat een aantal frequentiepatronen en een aantal CMYK-beelden voorgesteld in figuur A.12. Alle beelden bevatten 2560 × 2048 of 2048 × 2560 pixels. Merk op dat “N1-cmyk”, “N2-cmyk” en “N5-cmyk” exact dezelfde beelden zijn als respectievelijk “woman-cmyk”, “cafe-cmyk” en “bike-cmyk”.

N1-cmyk
2048 × 2560

N2-cmyk
2048 × 2560

N3-cmyk
2560 × 2048

N4-cmyk
2560 × 2048

N5-cmyk
2048 × 2560

N6-cmyk
2560 × 2048

N7-cmyk
2560 × 2048

N8-cmyk
2560 × 2048

Figuur A.12: De SCID CMYK-testbeelden van ISO.

Bijlage B

Afkortingen

Deze appendix vat alle gebruikte afkortingen en termen samen in een overzichtelijke lijst. Op het einde van elke lijn staat het paginanummer waar de corresponderende techniek of het gerefereerde concept beschreven wordt. De meeste termen en afkortingen komen uit het Engels en zijn onveranderd weergegeven.

ACB Associative Coder of Buyanovsky, 71

AEP Asymptotic Equipartition Property, 16

AM Amplitude Modulation, 153

AT Adaptive Template, 85

BTPC Binary Tree Predictive Coding, 112

BT-RC Binary Tree — Residual Coding, 201

BWT Burrows-Wheeler Transform, 71

BZIP Block-sorting file compressor, 71

BZIP2 Block-sorting file compressor version 2, 71

CALIC Context-based, Adaptive, Lossless Image Codec, 106

CCITT International Telegraph and Telephone Consultative Committee, 77

CMYK Cyan, Magenta, Yellow, and blacK, 252

CREW Compression with Reversible Embedded Wavelets, 119

CTW Context Tree Weighting, 67

DCT Discrete Cosine Transform, 101

DHPC Dynamic History Predictive Compression, 65

DMC Dynamic Markov Coding, 65

DMS	Discrete Memoryless Source, 9
DRC	Dispersed Reference Compression, 193
DWT	Discrete Wavelet Transform, 116
EBCOT	Embedded Block Coding with Optimized Truncation, 117
EZW	Embedded Zerotree Wavelet, 114
FCM	Finite Context Model, 54
FELICS	Fast Efficient Lossless Image Coding System, 102
FM	Frequency Modulation, 153
FO-RC	Fixed Order – Residual Coding, 201
FSM	Finite-State Machine, 65
G3	Group 3, 77
G4	Group 4, 77
GAP	Gradient-Adjusted Prediction, 108
GCR	Gray Color Replacement, 145
GZIP	Gnu ZIP, 70
HINT	Hierarchical Interpolation, 111
HMM	Hidden Markov Model, 81
HVS	Human Visual System, 8
IB-CALIC	Interband CALIC, 123
ICT	Irreversible Component Transformation, 123
IEC	International Electrotechnical Commission, 83
IEP	Inter-color Error Prediction, 141
ISO	International Organization for Standardization, 83
ITU-T	International Telecommunications Union — Telecommunications Standardization Sector, 83
JBIG	Joint Bi-level Image Experts Group, 83
JBIG1	Joint Bi-level Image Experts Group version 1, 83
JBIG2	Joint Bi-level Image Experts Group version 2, 88
JBIG-S	JBIG — Serial mode, 127
JPEG	Joint Photographic Experts Group, 101
JPEG2000	Joint Photographic Experts Group 2000, 114
JPEG-LS	JPEG — Lossless, 103
KLT	Karhunen-Loève Transform, 124
KLT-B	Karhunen-Loève Transform — on Blocks, 142

PPMC Prediction by Partial Matching — Method C, 64
PPMC' Prediction by Partial Matching — Method C', 64
PPMD+ Prediction by Partial Matching — Method D Plus, 64
PPMDE Prediction by Partial Matching — Method D/E, 126
PPMZ Prediction by Partial Matching — Method Z, 66
PPMZ2 Prediction by Partial Matching — Method Z version 2, 66
PPPM Prediction by Partial Precision Matching, 106
Q Q-coder, 44
QM QM-coder, 44
QoS Quality of Service, 115
Qx Qx-coder, 44
RCT Reversible Component Transformation, 123
RGB Red, Green, and Blue, 252
RLE Run-Length Encoding, 77
ROI Region of Interest, 115
SICLIC Simple Inter-Color Lossless Image Coder, 123
SLIC Segmentation-based Lossless Image Coding, 120
S+P Sequential and Predictive, 112
SPIHT Set Partitioning In Hierarchical Trees, 114
SPM Soft Pattern Matching, 90
SVD Singular Value Decomposition, 144
SZIP Block Sorting ZIP, 71
TIC Textual Image Compression, 87
TIFF Tag Image File Format, 89
T-PB TIFF — Packbits, 127
T-LZW TIFF — LZW, 127
UCM Universal Context Modelling, 105
VaR Value at Risk, 208
VW98 Volf-Willems 1998, 69
WFA Weighted Finiate Automata, 121
WTCQ Wavelet/Trellis Coded Quantization, 119

Bijlage C

Publicaties

Hierna volgt een overzicht van alle publicaties die verschenen zijn gedurende dit onderzoek en waaraan we hebben bijgedragen. De lijst is georganiseerd volgens: (i) publicaties in internationale tijdschriften, (ii) bijdragen op internationale conferenties en (iii) overige publicaties. De opsomming verloopt alfabetisch volgens de eerste auteur.

De publicaties waar in de loop van dit proefschrift naar gerefereerd wordt, zijn eveneens opgenomen in de bibliografie op het einde van dit proefschrift. De indices van de referenties in de loop van het proefschrift stemmen overeen met de indices van de bibiografie, maar niet met de indices van de hiernavolgende publicatielijst.

C.1 Publicaties in internationale tijdschriften

1. P. De Neve, K. Denecker, W. Philips, and I. Lemahieu. An advanced color representation for lossy compression of CMYK prepress images. *Computer Graphics Forum*, 19(1), pp. 3–12, March 2000.

2. K. Denecker, P. De Neve, S. Van Assche, R. Van de Walle, I. Lemahieu, and W. Philips. Psychovisual evaluation of lossy CMYK image compression for printing applications. *Computer Graphics Forum*, 21(1), pp. 5–17, March 2002.

3. K. Denecker, S. Van Assche, J. Crombez, R. Vander Vennet, and I. Lemahieu. Value-at-risk prediction using context modeling. *The European Physical Journal B*, 20(4), pp. 481–492, April 2001.

4. K. Denecker, S. Van Assche, P. De Neve, and I. Lemahieu. Context-based lossless halftone image compression. *Journal of Electronic Imaging*, 8(4), pp. 404–414, October 1999.

5. K. Denecker, D. Van De Ville, F. Habils, W. Meeus, M. Brunfaut, and I. Lemahieu. Design of an improved lossless halftone image compression codec. *Signal Processing: Image Communication*, 17(3), pp. 277–292, March 2002.

6. K. Denecker, J. Van Overloop, and F. Sommen. The general quadratic Radon transform. *Inverse Problems*, 14(3), pp. 615–633, June 1998.

7. W. Philips, K. Denecker, P. De Neve, and S. Van Assche. Lossless quantization of Hadamard transform coefficients. *IEEE Transactions on Image Processing*, 9(11), pp. 1995–1999, November 2000.

8. W. Philips, S. Van Assche, D. De Rycke, and K. Denecker. State-of-the-art techniques for lossless compression of 3D medical image sets. *Computerized Medical Imaging and Graphics*, 25(2), pp. 173–185, March 2001.

9. S. Van Assche, K. Denecker, and P. De Neve. Evaluation of lossless compression techniques for high-resolution RGB and CMYK color images. *Journal of Electronic Imaging*, 8(4), pp. 415–421, October 1999.

10. D. Van De Ville, K. Denecker, W. Philips, and I. Lemahieu. Nonlinear resampling for edge preserving moiré suppression. *Journal of Electronic Imaging*, 9(4), pp. 534–547, October 2000.

C.2 Bijdragen op internationale conferenties

1. P. De Neve, K. Denecker, and I. Lemahieu. A CIELab driven adaptive quantization scheme for DCT-based compression of CMYK images. In B. E. Rogowitz and T. N. Pappas, eds., *Human Vision and Electronic Imaging IV*, volume 3644 of *Proceedings of SPIE*, pp. 382–389. SPIE, IS&T, San Jose, CA, USA, January 1999.

2. P. De Neve, K. Denecker, W. Philips, and I. Lemahieu. Compressing CMYK images by removing color redundancy: A perceptual based quality evaluation. In J. Bares, ed., *Electronic Imaging: Processing, Printing, and Publishing in Color*, volume 3409 of *Proceedings of SPIE*, pp. 267–276. EUROPTO, Zürich, Switzerland, May 1998.

3. P. De Neve, W. Philips, K. Denecker, and I. Lemahieu. Introducing a decorrelated color space in the lossy compression of pre-press applications. In *The Fifth Color Imaging Conference: Color Science, Systems and Applications*, pp. 88–91. IS&T/SID, Scottsdale, AZ, USA, November 1997.

4. K. Denecker. Universal binary context modeling of image residue for lossless image compression. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 119–124. STW Technology Foundation, Mierlo, The Netherlands, November 1998.

5. K. Denecker and P. De Neve. A comparative study of lossless coding techniques for screened continuous-tone images. In B. Werner, ed., *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, volume IV, pp. 2941–2944. IEEE Computer Society Press, Munich, Germany, April 1997.

6. K. Denecker, P. De Neve, and I. Lemahieu. Binary tree context modeling of halftone images using a fast adaptive template selection scheme. In J. Bares, ed., *Electronic Imaging: Processing, Printing, and Publishing in Color*, volume 3409 of *Proceedings of SPIE*, pp. 230–241. EUROPTO, Zürich, Switzerland, May 1998.

7. K. Denecker, P. De Neve, and I. Lemahieu. Improved lossless halftone image coding using a fast adaptive context template selection scheme. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 541. IEEE Computer Society, Snowbird, UT, USA, March 1998.

8. K. Denecker, P. De Neve, B. Rogge, D. Van De Ville, and I. Lemahieu. Visually uniform near-lossless CMYK image compression. In W. Hahn, E. Walther-Klaus, and J. Knop, eds., *Euromedia*, pp. 59–63. Society for Computer Simulation International, Munich, Germany, April 1999.

9. K. Denecker and I. Lemahieu. Lossless colour image compression using inter-colour error prediction. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 95–100. STW Technology Foundation, Mierlo, The Netherlands, November 1996.

10. K. Denecker, S. Van Assche, and P. De Neve. CIELab-based near-lossless compression of prepress images. In G. B. Beretta and R. Eschbach, eds., *Color Imaging: Device-Independent Color, Color Hard Copy and Graphic Arts IV*, volume 3648 of *Proceedings of SPIE*, pp. 328–335. SPIE, IS&T, San Jose, CA, USA, January 1999.

11. K. Denecker, S. Van Assche, and I. Lemahieu. Binary tree context modeling of halftone images using a fast adaptive template selection scheme. In *Signal Processing Symposium*, pp. 75–78. IEEE Benelux, Leuven, Belgium, March 1998.

12. K. Denecker, S. Van Assche, and I. Lemahieu. A fast autocorrelation based context template selection scheme for lossless compression of halftone images. In G. B. Beretta and R. Eschbach, eds., *Color Imaging: Device Independent Color, Color Hardcopy, and Graphic Arts III*, volume 3300 of *Proceedings of SPIE*, pp. 262–272. SPIE, IS&T, San Jose, CA, USA, January 1998.

13. K. Denecker, S. Van Assche, W. Philips, and I. Lemahieu. Comparison of lossless coding techniques for screened continuous-tone images. In N. Ohta, ed., *Digital Compression Technologies and Systems for Video Communications*, volume 2952 of *Proceedings of SPIE*, pp. 122–132. Europto, Berlin, Germany, October 1996.

14. K. Denecker, S. Van Assche, W. Philips, and I. Lemahieu. State of the art concerning lossless medical image coding. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 129–136. STW Technology Foundation, Mierlo, The Netherlands, November 1997.

15. K. Denecker, S. Van Assche, W. Philips, and I. Lemahieu. Bit-oriented context modeling of the prediction error residue for lossless image compression. In M. H. Hamza, ed., *Proceedings of the IASTED International Conference Computer Graphics and Imaging*, pp. 138–141. IASTED, ACTA, Halifax, Nova Scotia, Canada, June 1998.

16. K. Denecker, D. Van De Ville, F. Habils, I. Lemahieu, and A. Munteanu. Software and hardware implementation of an improved lossless halftone image compression algorithm. In *International Congress on Imaging Science, Electronic Imaging*, volume 2, pp. 263–267. Antwerp, Belgium, September 1998.

17. K. Denecker, J. Van Overloop, and I. Lemahieu. An experimental comparison of several lossless image coders for medical images. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 435. IEEE Computer Society Press, Snowbird, UT, USA, March 1997.

18. K. Denecker, J. Van Overloop, and I. Lemahieu. An experimental comparison of several lossless image coders for medical images. In S. Hagerty and R. Renner, eds., *Proceedings of the Data Compression Industry Workshop*, pp. 67–76. Ball Aerospace & Technologies Corp., Snowbird, UT, USA, March 1997.

19. W. Philips and K. Denecker. Adaptive warped polynomial filtering of biomedical signals. In *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 235–242. STW Technology Foundation, Mierlo, The Netherlands, March 1995.

20. W. Philips and K. Denecker. A lossless version of the Hadamard transform. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 443–450. STW Technology Foundation, Mierlo, The Netherlands, November 1997.

21. W. Philips and K. Denecker. A new embedded lossless/quasi-lossless image coder based on the Hadamard transform. In *Proceedings of the IEEE International Conference on Image Processing*, volume I, pp. 667–670. Santa Barabara, CA, USA, October 1997.

22. W. Philips, K. Denecker, P. De Neve, and S. Van Assche. Very high quality compression of color and gray scale images: Techniques and applications. In *International Congress on Imaging Science, Electronic Imaging*, volume 2, pp. 259–263. Antwerp, Belgium, September 1998.

23. B. Rogge, I. Lemahieu, W. Philips, K. Denecker, P. De Neve, and S. Van Assche. Region of interest based progressive transmission of greyscale images across the Internet. In G. B. Beretta and R. Eschbach, eds., *Color*

Imaging: Device-Independent Color, Color Hard Copy and Graphic Arts IV, volume 3648 of *Proceedings of SPIE*, pp. 365–372. SPIE, IS&T, San Jose, CA, USA, January 1999.

24. B. Rogge, I. Lemahieu, W. Philips, K. Denecker, P. De Neve, and S. Van Assche. Region of interest based progressive transmission of greyscale images across the Internet. In W. Hahn, E. Walther-Klaus, and J. Knop, eds., *Euromedia*, pp. 28–32. Society for Computer Simulation International, Munchen, Germany, April 1999.

25. B. Rogge, I. Lemahieu, W. Philips, K. Denecker, S. Van Assche, and P. De Neve. Region of interest based progressive transmission of greyscale images across the Internet. In G. Joubert and E. D'Hollander, eds., *High Performance Computing: Biomedical Applications and Parallel Architectures*, pp. 79–89. Communication & Cognition, Clausthal, Germany, January 1999.

26. B. Rogge, I. Lemahieu, W. Philips, S. Van Assche, K. Denecker, and P. De Neve. ROI-based progressive transmission of medical images across the internet. In L. J. Lambrecht, ed., *European Symposium on Clinical Imaging and Networking*, p. 44. Antwerp, Belgium, April 1998.

27. S. Van Assche, K. Denecker, J. Crombez, and W. Philips. Value-at-Risk prediction using context modeling. In K. Klemm and P. Alstrøm, eds., *Applications of Physics in Financial Analysis - 2*, volume 24E, p. 14. European Physical Society, Liège, Belgium, July 2000.

28. S. Van Assche, K. Denecker, J. Crombez, R. Vander Vennet, and I. Lemahieu. Value-at-Risk estimation using context modeling. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 537–548. STW Technology Foundation, Mierlo, The Netherlands, December 2000.

29. S. Van Assche, K. Denecker, W. Philips, and I. Lemahieu. Adaptation of the CCITT Group 4 fax standard to screened images. In *International Congress on Imaging Science, Electronic Imaging*, volume 2, pp. 294–297. Antwerp, Belgium, September 1998.

30. S. Van Assche, K. Denecker, W. Philips, and I. Lemahieu. Lossless compression of three-dimensional medical images. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 549–553. STW Technology Foundation, Mierlo, The Netherlands, November 1998.

31. S. Van Assche, K. Denecker, W. Philips, and I. Lemahieu. A comparison of lossless compression techniques for prepress color images. In K. Aizawa, ed., *Visual Communications and Image Processing*, volume 3653 of *Proceedings of SPIE*, pp. 1376–1383. SPIE, IS&T, San Jose, CA, USA, January 1999.

32. S. Van Assche, K. Denecker, W. Philips, and I. Lemahieu. Lossless compression of color images using color decorrelation. In *Proceedings of the World*

Multiconference on Systems, Cybernetics and Informatics, pp. 250–256. Orlando, FL, USA, August 1999.

33. S. Van Assche, K. Denecker, W. Philips, and I. Lemahieu. State-of-the-art concerning lossless compression of three-dimensional medical images. In *Fifth Conference of the European Society for Engineering and Medicine*, pp. 573–574. Barcelona, Spain, May 1999.

34. D. Van De Ville, K. Denecker, and I. Lemahieu. Non-linear filtering for moiré suppression in gravure printing. In *International Congress on Imaging Science, Electronic Imaging*, volume 2, pp. 252–256. Antwerp, Belgium, September 1998.

35. D. Van De Ville, K. Denecker, W. Philips, and I. Lemahieu. Non-linear resampling for both moiré suppression and edge preservation. In G. B. Beretta and R. Eschbach, eds., *Color Imaging: Device-Independent Color, Color Hard Copy and Graphic Arts IV*, volume 3648 of *Proceedings of SPIE*, pp. 432–441. SPIE, IS&T, San Jose, CA, USA, January 1999.

36. D. Van De Ville, K. Denecker, W. Philips, and I. Lemahieu. An overview of resampling filters for halftoned images. In G. Joubert and E. D'Hollander, eds., *High Performance Computing: Biomedical Applications and Parallel Architectures*, pp. 91–106. Communication & Cognition, Clausthal, Germany, January 1999.

37. D. Van De Ville, K. Denecker, W. Philips, and I. Lemahieu. Gabor spectrogram based risk of aliasing. In N. Mastorakis, ed., *Fourth World Multi-Conference on: Circuits, Systems, Communications and Computers*, pp. 3631–3636. World Scientific and Engineering Society, July 2000.

38. D. Van De Ville, K. Denecker, W. Philips, and I. Lemahieu. An objective quality measure for the assessment of aliasing and sharpness in halftones. In J. Blanc-Talon and D. Popescu, eds., *Proceedings of the Second International Symposium on Intelligent Vision Systems*, pp. 23–30. The International Institute for Advanced Studies in Systems Research and Cybernetics, Baden-Baden, Germany, August 2000.

39. D. Van De Ville, K. Denecker, W. Philips, and I. Lemahieu. On the measurement of aliasing and sharpness in halftones. In F. Broeckx and L. Pauwels, eds., *Euromedia*, pp. 311–316. Society for Computer Simulation International, Antwerp, Belgium, May 2000.

40. D. Van De Ville, K. Denecker, W. Philips, and I. Lemahieu. A quality measure for halftones obtained by linear and nonlinear resampling. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 549–556. STW Technology Foundation, Mierlo, The Netherlands, December 2000.

41. D. Van De Ville, K. Denecker, W. Philips, and I. Lemahieu. *Signal Processing, Communications and Computer Science*, Chapter Gabor spectrogram

based risk of aliasing, pp. 37–42. Electrical and Computer Engineering Series. World Scientific Engineering Society Press, 2000.

42. J. Van Overloop, W. Van De Sype, K. Denecker, P. De Neve, E. Sundermann, and I. Lemahieu. An experimental comparison of 2D and 3D wavelet image compression methods for medical image sets. In Y. Kim and S. K. Mun, eds., *Medical Imaging, Image Display*, volume 3335 of *Proceedings of SPIE*, pp. 348–358. SPIE, San Diego, CA, USA, February 1998.

C.3 Overige publicaties

1. K. Denecker. Context modelling of halftone images. In *First FTW PhD Symposium*, pp. 1–2. Ghent University, Ghent, Belgium, December 2000. Paper number 60.

2. D. Van De Ville, K. Denecker, W. Philips, and I. Lemahieu. *Imaging and Vision Systems: Theory, Assessment and Applications*, Chapter Joint edge-preserving and moiré-suppressing image resampling using the discrete Gabor transform. Advances in Computation: Theory and Practice. NOVA Science Books, Huntington, NY, USA, October 2000.

Bibliografie

[1] J. Åberg, Y. M. Shtarkov, and B. J. M. Smeets. Towards understanding and improving escape probabilities in PPM. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 22–31. IEEE Computer Society Press, Snowbird, UT, USA, March 1997.

[2] N. Abramson. *Information theory and coding*. McGraw-Hill, New York, NY, USA, 1963.

[3] C. Alexander. *The handbook of risk management and analysis*, Chapter Volatility and Correlation Forecasting. John Wiley & Sons, West Sussex, United Kingdom, 1996.

[4] R. Ansari, N. Memon, and E. Ceran. Near-lossless image compression techniques. *Journal of Electronic Imaging*, 7(3), pp. 486–494, July 1998.

[5] R. Arnold and T. Bell. A corpus for the evaluation of lossless compression algorithms. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 201–210. IEEE Computer Society Press, Snowbird, UT, USA, March 1997.

[6] R. B. Arps and T. K. Truong. Comparison of international standards for lossless still image compression. *Proceedings of the IEEE*, 82(6), pp. 889–899, June 1994.

[7] R. B. Arps, T. K. Truong, D. J. Lu, R. C. Pasco, and T. D. Friedman. A multipurpose VLSI chip for adaptive data compression of bilevel images. *IBM Journal of Research and Development*, 32(6), pp. 775–795, November 1988.

[8] J. Augustine, W. Fen, A. Makur, and J. Jacob. Switching theoretic approach to image compression. *Signal Processing*, 44(2), pp. 243–246, June 1995.

[9] Bank for International Settlements. Capital requirements and bank behaviour: The impact of the Basel accord. Working Paper No. 1, 1999.

[10] Bank for International Settlements. A new capital adequacy framework. No. 50, 1999.

[11] Bank for International Settlements, Basel, Switzerland. *Stress Testing by Large Financial Institutions: Current Practice and Aggregation Issues*, April 2000.

[12] R. Barequet and M. Feder. SICLIC: A simple inter-color lossless image coder. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 501–510. IEEE Computer Society Press, Snowbird, UT, USA, March 1999.

[13] T. Bell and A. Moffat. A note on the DMC data compression scheme. *Computer Journal*, 32(1), pp. 16–20, February 1989.

[14] T. C. Bell, J. G. Cleary, and I. H. Witten. *Text compression*. Advanced reference series Computer Science. Prentice Hall, Englewood Cliffs, NJ, USA, 1990.

[15] R. E. Blahut. *Theory and practice of error control codes*. Addison-Wesley, Reading, MA, USA, 1983.

[16] C. Bloom. LZP: A new data compression algorithm. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 425. IEEE Computer Society Press, Snowbird, UT, USA, March 1996.

[17] C. Bloom. Solving the problems of context modeling, March 1998. http://www.cbloom.com/papers/.

[18] D. Bodson, S. J. Urban, A. R. Deutermann, and C. E. Clarke. Measurement of data compression in advanced Group 4 facsimile systems. *Proceedings of the IEEE*, 73(4), pp. 731–739, April 1985.

[19] M. Boliek, M. J. Gormish, E. L. Schwartz, and A. Keith. Decoding compression with reversible embedded wavelets (CREW) codestreams. *Journal of Electronic Imaging*, 7(3), pp. 402–409, July 1998.

[20] L. Brillouin. *Science and information theory*. Academic Press, New York, NY, USA, 1962.

[21] S. Bunton. An executable taxonomy of on-line modeling algorithms. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 42–51. IEEE Computer Society Press, Snowbird, UT, USA, March 1997.

[22] S. Bunton. Generalization and improvement to PPM's "blending". In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 426. IEEE Computer Society Press, Snowbird, UT, USA, March 1997.

[23] S. Bunton. A percolating state selector for suffix-tree context models. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 32–41. IEEE Computer Society Press, Snowbird, UT, USA, March 1997.

[24] M. Burrows and D. J. Wheeler. A block-sorting lossless data compression algorithm. Technical Report SRC-124, Digital Equipment Corporation, Palo Alto, CA, USA, May 1994.

[25] J. Y. Campbell. Stock returns and the term structure. *Journal of Financial Economics*, 18(2), pp. 373–399, June 1987.

[26] J. Y. Campbell, A. W. Lo, and A. C. MacKinlay. *The econometrics of financial markets*. Princeton University Press, Princeton, NJ, USA, 1997.

[27] J. Y. Campbell and R. J. Shiller. Stock-prices, earnings, and expected dividends. *Journal of Finance*, 43(3), pp. 661–676, July 1988.

[28] A. Campos. Implementing PPMC with hash tables, March 2000. http://www.arturocampos.com/ac_ppmc.html.

[29] R. M. Capocelli and A. De Santis. *Image and text compression*, Chapter Variations on a Theme by Gallager, pp. 181–213. Kluwer Academic Pubishers, 1992.

[30] R. M. Capocelli, R. Giancarlo, and I. J. Taneja. Bounds on the redundancy of Huffman codes. *IEEE Transactions on Information Theory*, IT-32(6), pp. 854–857, November 1986.

[31] J. Capon. A probabilistic model for run-length coding of pictures. *IRE Transactions on Information Theory*, IT-5(4), pp. 157–163, December 1959.

[32] B. Carpentieri, M. J. Weinberger, and G. Seroussi. Lossless compression of continuous-tone images. *Proceedings of the IEEE*, 88(11), pp. 1797–1809, November 2000.

[33] A. K. Chaudhary, J. Augustine, and J. Jacob. Lossless compression of images using logic minimization. In *Proceedings of the IEEE International Conference on Image Processing*, volume I, pp. 77–80. IEEE, Lausanne, Switzerland, September 1996.

[34] N.-F. Chen, R. Roll, and S. A. Ross. Economic forces and the stock-market. *Journal of Business*, 59(3), pp. 383–403, July 1986.

[35] D. Chevion, E. D. Karnin, and E. Walach. High efficiency, multiplication free approximation of arithmetic coding. In J. A. Storer and J. H. Reif, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 43–52. IEEE Computer Society Press, Snowbird, UT, USA, April 1991.

[36] P. F. Christoffersen, F. X. Diebold, and T. Schuermann. Horizon problems and extreme events in financial risk management. *Federal Reserve Bank of New York Economic Policy Review*, 4(3), pp. 109–118, October 1998.

[37] R. J. Clarke. *Digital compression of still images and video*. Academic Press, London, United Kingdom, 1995.

[38] J. G. Cleary, W. J. Teahan, and I. H. Witten. Unbounded length contexts for PPM. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 52–61. IEEE Computer Society Press, Snowbird, UT, USA, March 1995.

[39] J. G. Cleary and I. H. Witten. Data compression using adaptive coding and partial string matching. *IEEE Transactions on Communications*, 32(4), pp. 396–402, April 1984.

[40] G. V. Cormack and R. N. S. Horspool. Data compression using dynamic Markov modelling. *Computer Journal*, 30(6), pp. 541–550, December 1987.

[41] T. M. Cover and R. C. King. A convergent gambling estimate of the entropy of English. *IEEE Transactions on Information Theory*, IT-24(4), pp. 413–421, July 1978.

[42] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley Series in Telecommunications. Wiley, New York, NY, USA, 1991.

[43] K. Culik and V. Valenta. Finite automata based compression of bi-level and simple color images. *Computers & Graphics*, 21(1), pp. 61–68, January 1997.

[44] R. P. Damodare, J. Augustine, and J. Jacob. Lossless and lossy image compression using Boolean minimization. *Sadhana-Academy Proceedings in Engineering Sciences*, 21, pp. 55–64, February 1996. Part 1.

[45] J. Danielsson, C. G. de Vries, and B. N. Jørgensen. The value of value at risk: Statistical, financial, and regulatory considerations – summary of presentation. *Federal Reserve Bank of New York Economic Policy Review*, 4(3), pp. 107–108, October 1998.

[46] G. De Micheli. *Synthesis and Optimization of Digital Circuits*. McGraw-Hill, 1994.

[47] K. Denecker. Universal binary context modeling of image residue for lossless image compression. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 119–124. STW Technology Foundation, Mierlo, The Netherlands, November 1998.

[48] K. Denecker. Context modelling of halftone images. In *First FTW PhD Symposium*, pp. 1–2. Ghent University, Ghent, Belgium, December 2000. Paper 60.

[49] K. Denecker and P. De Neve. A comparative study of lossless coding techniques for screened continuous-tone images. In B. Werner, ed., *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, volume IV, pp. 2941–2944. IEEE Computer Society Press, Munich, Germany, April 1997.

[50] K. Denecker, P. De Neve, and I. Lemahieu. Binary tree context modeling of halftone images using a fast adaptive template selection scheme. In J. Bares, ed., *Electronic Imaging: Processing, Printing, and Publishing in Color*, volume 3409 of *Proceedings of SPIE*, pp. 230–241. EUROPTO, Zürich, Switzerland, May 1998.

[51] K. Denecker, P. De Neve, and I. Lemahieu. Improved lossless halftone image coding using a fast adaptive context template selection scheme. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 541. IEEE Computer Society, Snowbird, UT, USA, March 1998.

[52] K. Denecker, P. De Neve, B. Rogge, D. Van De Ville, and I. Lemahieu. Visually uniform near-lossless CMYK image compression. In W. Hahn, E. Walther-Klaus, and J. Knop, eds., *Euromedia*, pp. 59–63. Society for Computer Simulation International, Munich, Germany, April 1999.

[53] K. Denecker, P. De Neve, S. Van Assche, and R. Van de Walle. Psychovisual evaluation of lossy CMYK image compression for printing applications. *Computer Graphics Forum*, 21(1), pp. 5–17, March 2002.

[54] K. Denecker and I. Lemahieu. Lossless colour image compression using inter-colour error prediction. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 95–100. STW Technology Foundation, Mierlo, The Netherlands, November 1996.

[55] K. Denecker, S. Van Assche, J. Crombez, R. Vander Vennet, and I. Lemahieu. Value-at-risk prediction using context modeling. *The European Physical Journal B*, 20(4), pp. 481–492, April 2001.

[56] K. Denecker, S. Van Assche, and P. De Neve. CIELab-based near-lossless compression of prepress images. In G. B. Beretta and R. Eschbach, eds., *Color Imaging: Device-Independent Color, Color Hard Copy and Graphic Arts IV*, volume 3648 of *Proceedings of SPIE*, pp. 328–335. SPIE, IS&T, San Jose, CA, USA, January 1999.

[57] K. Denecker, S. Van Assche, P. De Neve, and I. Lemahieu. Context-based lossless halftone image compression. *Journal of Electronic Imaging*, 8(4), pp. 404–414, October 1999.

[58] K. Denecker, S. Van Assche, and I. Lemahieu. Binary tree context modeling of halftone images using a fast adaptive template selection scheme. In *Signal Processing Symposium*, pp. 75–78. IEEE Benelux, Leuven, Belgium, March 1998.

[59] K. Denecker, S. Van Assche, and I. Lemahieu. A fast autocorrelation based context template selection scheme for lossless compression of halftone images. In G. B. Beretta and R. Eschbach, eds., *Color Imaging: Device Independent Color, Color Hardcopy, and Graphic Arts III*, volume 3300 of *Proceedings of SPIE*, pp. 262–272. SPIE, IS&T, San Jose, CA, USA, January 1998.

[60] K. Denecker, S. Van Assche, W. Philips, and I. Lemahieu. Comparison of lossless coding techniques for screened continuous-tone images. In N. Ohta, ed., *Digital Compression Technologies and Systems for Video Communications*, volume 2952 of *Proceedings of SPIE*, pp. 122–132. Europto, Berlin, Germany, October 1996.

[61] K. Denecker, S. Van Assche, W. Philips, and I. Lemahieu. State of the art concerning lossless medical image coding. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 129–136. STW Technology Foundation, Mierlo, The Netherlands, November 1997.

[62] K. Denecker, S. Van Assche, W. Philips, and I. Lemahieu. Bit-oriented context modeling of the prediction error residue for lossless image compression. In M. H. Hamza, ed., *Proceedings of the IASTED International Conference Computer Graphics and Imaging*, pp. 138–141. IASTED, ACTA, Halifax, Nova Scotia, Canada, June 1998.

[63] K. Denecker, D. Van De Ville, F. Habils, I. Lemahieu, and A. Munteanu. Software and hardware implementation of an improved lossless halftone image compression algorithm. In *International Congress on Imaging Science, Electronic Imaging*, volume 2, pp. 263–267. Antwerp, Belgium, September 1998.

[64] K. Denecker, D. Van De Ville, F. Habils, W. Meeus, M. Brunfaut, and I. Lemahieu. Design of an improved lossless halftone image compression codec. *Signal Processing: Image Communication*, 17(3), pp. 277–292, March 2002.

[65] K. Denecker, J. Van Overloop, and I. Lemahieu. An experimental comparison of several lossless image coders for medical images. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 435. IEEE Computer Society Press, Snowbird, UT, USA, March 1997.

[66] K. Denecker, J. Van Overloop, and I. Lemahieu. An experimental comparison of several lossless image coders for medical images. In S. Hagerty and R. Renner, eds., *Proceedings of the Data Compression Industry Workshop*, pp. 67–76. Ball Aerospace & Technologies Corp., Snowbird, UT, USA, March 1997.

[67] K. Dowd. *Beyond Value at Risk: The New Science of Risk Management*. John Wiley & Sons, New York, NY, USA, 1998.

[68] D. L. Duttweiler and C. Chamzas. Probability estimation in arithmetic and adaptive-huffman entropy coders. *IEEE Transactions on Image Processing*, 4(3), pp. 237–246, March 1995.

[69] P. Elias. Universal codeword sets and representations of the integers. *IEEE Transactions on Information Theory*, 21(2), pp. 194–203, March 1975.

[70] C. Eliezer. Color screening technology: A tutorial on the basic issues. *The Seybold Report on Desktop Publishing*, 6(2), pp. 3–25, October 1991.

[71] E. J. Elton and M. J. Gruber. *Modern portfolio theory and investment analysis*. John Wiley & Sons, New York, NY, USA, 1995.

[72] T. Endoh and Y. Yamakazi. Progressive coding scheme for multilevel images. In *Proceedings of the Picture Coding Symposium*, pp. 21–22. SPIE, Tokyo, Japan, 1986.

[73] N. Faller. An adaptive system for data compression. In *Record of the 7th Asilomar Conference on Circuits, Systems, and Computers*, pp. 593–597. IEEE Press, Piscataway, NJ, USA, 1973.

[74] R. M. Fano. The transmission of information. Technical Report 65, Research Laboratory of Electronics, MIT, Cambridge, MA, USA, 1949.

[75] P. Fenwick. Symbol ranking text compressors: Review and implementation. *Software — Practice and Experience*, 28(5), pp. 547–559, April 1998.

[76] W. E. Ferson and C. R. Harvey. The variation of economic risk premiums. *Journal of Political Economics*, 99(2), pp. 385–415, April 1991.

[77] E. R. Fiala and D. H. Greene. Data compression with finite windows. *Communications of the ACM*, 32(4), pp. 490–505, April 1989.

[78] M. Fisz. *Probability Theory and Mathematical Statistics*. Wiley, New York, NY, USA, 1963.

[79] S. Forchhammer and K. S. Jensen. Data compression of scanned halftone images. *IEEE Transactions on Communications*, 42, pp. 1881–1893, February 1994.

[80] J. A. Frankel. The internationalization of equity markets. *The University of Chicago Press*, 1994.

[81] R. G. Gallager. Variations on a theme by Huffman. *IEEE Transactions on Information Theory*, 24(6), pp. 668–674, November 1978.

[82] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1992.

[83] S. W. Golomb. Run-length encodings. *IEEE Transactions on Information Theory*, 12(3), pp. 399–401, May 1966.

[84] F. Gray. Pulse code communication. U.S. Patent 2 632 058, March 1953.

[85] M. Guazzo. A general minimum-redundancy source-coding algorithm. *IEEE Transactions on Information Theory*, IT-26(1), pp. 15–25, January 1980.

[86] R. Hamming. Error detecting and error correcting codes. *Bell Systems Technical Journal*, 29, pp. 147–160, 1950.

[87] H.-M. Hang and J. W. Woods, eds. *Handbook of visual communications*. Academic Press, San Diego, CA, USA, 1995.

[88] C. R. Harvey. The real term structure and consumption growth. *Journal of Financial Economics*, 22(2), pp. 305–333, December 1988.

[89] B. G. Haskell, P. G. Howard, Y. A. LeCun, A. Puri, J. Ostermann, M. R. Civanlar, L. Rabiner, L. Bottou, and P. Haffner. Image and video coding—emerging standards and beyond. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(7), pp. 814–837, November 1998.

[90] R. A. Haugen and N. L. Baker. Commonality in the determinants of expected stock returns. *Journal of Financial Economics*, 41(3), pp. 401–439, July 1996.

[91] G. Held and T. R. Marshall. *Data and image compression: tools and techniques*. John Wiley & Sons, 4th edition, 1996.

[92] R. Hill. *A first course in coding theory*. Oxford University Press, Oxford, United Kingdom, 1990.

[93] D. Hirschberg and D. Lelewer. Efficient decoding of prefix codes. *Communications of the ACM*, 33(4), pp. 449–459, April 1990.

[94] D. G. Hoffman, D. A. Leanord, C. C. Lidner, K. T. Phelps, and C. A. Rodger. *Coding theory: the essentials*. Marcel Dekker, New York, NY, USA, 1991.

[95] R. Hoffman. *Data compression in digital systems*. Chapman & Hall, New York, NY, USA, 1996.

[96] R. N. Horspool and G. V. Cormack. Dynamic Markov modelling—a prediction technique. In *Proceedings 19th Hawaii International Conference on System Sciences*, volume II, pp. 700–707. Honolulu, HA, USA, January 1986.

[97] P. G. Howard. The design and analysis of efficient lossless data compression systems. Technical Report CS-93-28, Brown University, Dept. of Computer Science, 1993.

[98] P. G. Howard. Text image compression using soft pattern matching. *Computer Journal*, 40(2–3), pp. 146–156, 1997.

[99] P. G. Howard, F. Kossentini, B. Martins, S. Forchhammer, and W. J. Rucklidge. The emerging JBIG2 standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(7), pp. 838–848, November 1998.

[100] P. G. Howard and J. S. Vitter. Error modeling for hierarchical lossless image compression. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 269–278. IEEE Computer Society Press, Snowbird, UT, USA, March 1992.

[101] P. G. Howard and J. S. Vitter. *Image and text compression*, Chapter Practical Implementations of Arithmetic Coding, pp. 85–112. Kluwer Academic Publishers, 1992.

[102] P. G. Howard and J. S. Vitter. New methods for lossless image compression using arithmetic coding. *Information Processing and Management*, 28(6), pp. 765–779, November 1992.

[103] P. G. Howard and J. S. Vitter. Fast and efficient lossless image compression. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 351–360. IEEE Computer Society Press, Snowbird, UT, USA, March 1993.

[104] P. G. Howard and J. S. Vitter. Arithmetic coding for data compression. *Proceedings of the IEEE*, 82(6), pp. 857–865, June 1994.

[105] D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40, pp. 1098–1101, 1952.

[106] R. W. G. Hunt. *The reproduction of colour*. Fountain Press, 5th edition, 1995.

[107] R. Hunter and H. Robinson. International digital facsimile coding standards. *Proceedings of the IEEE*, 68(7), pp. 854–867, July 1980.

[108] S. Inglis. *Lossless Document Image Compression*. Ph.D. thesis, University of Waikato, Hamilton, New Zealand, March 1999.

[109] International Organization for Standardization ISO. *LOCO-A: An Arithmetic Coding Extension of LOCO-I*, June 1996. ISO/IEC JTC1/SC29/WG1 Doc. N342.

[110] International Organization for Standardization (ISO). *Graphic Technology – Prepress digital data exchange – CMYK standard color image data (CMYK/SCID)*, 1997.

[111] International Telecommunication Union (ITU-T). *Lossless and Near-Lossless Compression of Continuous-Tone Still Images: Baseline*, 2000. ITU-T Recommendation T.87 — ISO/IEC International Standard 14495-1:2000.

[112] International Telecommunication Union, Telecommunication standardization sector (ITU-T). *Standardization of Group 3 Facsimile Apparatus for Document Transmission*, 1980. CCITT Recommendation T.4.

[113] International Telecommunication Union, Telecommunication standardization sector (ITU-T). *Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus*, 1984. CCITT Recommendation T.6.

[114] J. P. Morgan/Reuters, New York, NY, USA. *RiskMetrics – Technical Document*, 4th edition, 1996.

[115] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice-Hall International Editions, 1989.

[116] E. T. Jaynes. *Papers on Probability, Statistics and Statistical Physics*. Reidel, Dordrecht, The Netherlands, 1982.

[117] F. Jelinek and K. S. Schneider. On variable-length-to-block coding. *IEEE Transactions on Information Theory*, IT-18(6), pp. 765–774, November 1972.

[118] I. JTC1-SC29-WG1. *Information Technology — Coded Representation of Picture and Audio Information — Progressive Bi-Level Image Compression*, 1993. ITU-T Recommendation T.82 — ISO/IEC International Standard 11544:1993.

[119] I. JTC1-SC29-WG1. *Digital Compression and Coding of Continuous-Tone Still Images*, 1994. ITU-T Recommendation T.81 — ISO/IEC International Standard 10918-1:1994.

[120] I. JTC1-SC29-WG1. *Information Technology — Coded Representation of Picture and Audio Information — Lossy/Lossless coding of Bi-Level Images*, 2001. ITU-T Recommendation T.82 — ISO/IEC International Standard 14492.

[121] I. JTC1-SC29-WG1. *Information Technology — JPEG 2000 Image Coding System — Part 1: Core Coding System*, 2001. ITU-T Recommendation T.800 — ISO/IEC International Standard 15444-1.

[122] F. A. Kampf. Performance as a function of compression. *IBM Journal of Research and Development*, 42(6), pp. 759–766, November 1998.

[123] H. R. Kang. *Digital Color Halftoning*. SPIE/IEEE Press, 1999.

[124] J. Karush. A simple proof of an inequality of McMillan. *IRE Transactions on Information Theory*, 7, p. 118, 1961.

[125] L. Ke and M. W. Marcellin. Near-lossless image compression: Minimum-entropy, constrained-error DPCM. *IEEE Transactions on Image Processing*, 7(2), pp. 225–228, February 1998.

[126] D. Kersten. Predictability and redundancy of natural images. *Journal of the Optical Society of America A — Optics Image Science and Vision*, 4(12), pp. 2395–2400, December 1987.

[127] Y. S. Kim and W. A. Pearlman. Lossless volumetric image compression. In *Applications of Digital Image Processing XXII*, volume 3808 of *Proceedings of SPIE*, pp. 305–312. October 1999.

[128] D. E. Knuth. Dynamic Huffman coding. *Journal of Algorithms*, 6(2), pp. 163–180, 1985.

[129] R. Krichevsky. *Universal compression and retrieval*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1994.

[130] G. R. Kuduvalli and R. M. Rangayyan. Performance analysis of reversible image compression techniques for high-resolution digital teleradiology. *IEEE Transactions on Medical Imaging*, 11(3), pp. 430–445, September 1992.

[131] R. Kuhn and R. De Mori. A cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6), pp. 570–583, June 1990.

[132] G. G. Langdon, Jr. An introduction to arithmetic coding. *IBM Journal of Research and Development*, 28(2), pp. 135–149, March 1984.

[133] G. G. Langdon, Jr. Sunset: A hardware oriented algorithm for lossless compression of gray scale images. In *Image Capture, Formatting, and Display*, volume 1444 of *Proceedings of SPIE*, pp. 272–282. March 1991.

[134] G. G. Langdon, Jr. and C. A. Haidinyak. Context-dependent distribution shaping and parameterization for lossless image compression. In A. G. Tescher, ed., *Applications of Digital Image Processing XVII*, volume 2298 of *Proceedings of SPIE*, pp. 62–70. September 1994.

[135] G. G. Langdon, Jr. and C. A. Haidinyak. Experiments with lossless and virtually lossless image compression algorithms. In M. Rabbani, E. J. Delp, and S. A. Rajala, eds., *Still-Image Compression*, volume 2418 of *Proceedings of SPIE*, pp. 21–27. SPIE, February 1995.

[136] G. G. Langdon, Jr. and J. Rissanen. Compression of black-white images with arithmetic coding. *IEEE Transactions on Communications*, COM-29(6), pp. 858–867, June 1981.

[137] G. G. Langdon, Jr. and J. J. Rissanen. A simple general binary source code. *IEEE Transactions on Information Theory*, 28(5), pp. 800–803, September 1982.

[138] J. Liang and T. D. Tran. Fast muliplierless approximations of the DCT with the lifting scheme. *IEEE Transactions on Signal Processing*, 49(12), pp. 3032–3044, December 2001.

[139] S. Lin and D. J. Costello, Jr. *Error control coding: fundamentals and applications*. Prentice Hall, Englewood Cliffs, NJ, USA, 1983.

[140] J. A. Lopez. Methods for evaluating value-at-risk estimates. *Federal Reserve Bank of New York Economic Policy Review*, 4(3), pp. 119–124, October 1998.

[141] J. A. Lopez. Methods for evaluating value-at-risk estimates. *Federal Reserve Bank of San Francisco Economic Review*, (2), pp. 3–17, May 1999.

[142] M. W. Marcellin, M. J. Gormish, A. Bilgin, and M. P. Boliek. An overview of JPEG-2000. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 523–541. IEEE Computer Society Press, Snowbird, UT, USA, March 2000.

[143] K. M. Marks. A JBIG-ABIC compression engine for digital document processing. *IBM Journal of Research and Development*, 42(6), pp. 753–758, November 1998.

[144] G. N. N. Martin. Range encoding: An algorithm for removing redundancy from a digitised message. In *Proceedings Video and Data Recording Conference*. Southhampton, United Kingdom, 1979.

[145] B. Martins and S. Forchhammer. Bi-level image compression with tree coding. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 270–279. IEEE Computer Society Press, Snowbird, UT, USA, March 1996.

[146] B. Martins and S. Forchhammer. Lossy/lossless coding of bi-level images. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 454. IEEE Computer Society Press, Snowbird, UT, USA, March 1997.

[147] B. Martins and S. Forchhammer. Tree coding of bilevel images. *IEEE Transactions on Image Processing*, 7(4), pp. 517–528, April 1998.

[148] B. Martins and S. Forchhammer. Lossless, near-lossless, and refinement coding of bilevel images. *IEEE Transactions on Image Processing*, 8(5), pp. 601–613, May 1999.

[149] B. Martins and S. Forchhammer. Halftone coding with JBIG2. *Journal of Electronic Imaging*, 9(1), pp. 52–60, January 2000.

[150] P. Mateu-Villarroya and J. Prades-Nebot. Lossless image compression using ordered binary-decision diagrams. *Electronics Letters*, 37(3), pp. 162–163, February 2001.

[151] B. McMillan. Two inequalities implied by unique decipherability. *IEEE Transactions on Information Theory*, 2, pp. 115–116, March 1956.

[152] N. Memon, D. L. Neuhoff, and S. Shende. An analysis of some common scanning techniques for lossless image coding. *IEEE Transactions on Image Processing*, 9(11), pp. 1837–1848, November 2000.

[153] N. Memon and K. Sayood. Lossless image compression: A comparative study. In M. Rabbani, E. J. Delp, and S. A. Rajala, eds., *Still-Image Compression*, volume 2418 of *Proceedings of SPIE*, pp. 8–20. March 1995.

[154] N. D. Memon and K. Sayood. A taxonomy for lossless image compression. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 526. IEEE Computer Society Press, Snowbird, UT, USA, March 1994.

[155] N. D. Memon, K. Sayood, and S. S. Magliveras. Lossless image compression with a codebook of block scans. *IEEE Journal on Selected Areas in Communications*, 13(1), pp. 24–30, January 1995.

[156] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of applied cryptography*. CRC Press, Boca Raton, FL, USA, 1996.

[157] J. L. Mitchell and W. B. Pennebaker. Software implementations of the Q-coder. *IBM Journal of Research and Development*, 32(6), pp. 753–774, November 1988.

[158] A. Moffat. A note on the PPM data compression algorithm. Research Report 88/7, Department of Computer Science, University of Melbourne, Parkville, Victoria, Australia, 1988.

[159] A. Moffat. Implementing the PPM data compression scheme. *IEEE Transactions on Communications*, 38(11), pp. 1917–1921, November 1990.

[160] A. Moffat. Two-level context based compression of binary images. In J. A. Storer and J. H. Reif, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 382–391. IEEE Computer Society Press, Snowbird, UT, USA, April 1991.

[161] A. Moffat, R. M. Neal, and I. H. Witten. Arithmetic coding revisited. In J. A. Storer and J. H. Reif, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 202–211. IEEE Computer Society Press, Snowbird, UT, USA, March 1995.

[162] A. Moffat, R. M. Neal, and I. H. Witten. Arithmetic coding revisited. *ACM Transactions on Information Systems*, 16(3), pp. 256–294, July 1998.

[163] A. N. Moffat, N. B. Sharman, I. H. Witten, and T. C. Bell. An empirical evaluation of coding methods for multi-symbol alphabets. In J. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 108–117. IEEE Computer Society Press, Snowbird, UT, USA, March 1993.

[164] J. D. Murray and W. van Ryper. *Encyclopedia of Graphics File Formats*. O'Reilly, Sebastopol, CA, USA, 1994.

[165] M. Nelson. *The data compression book*. M&T Books, 2nd edition, 1995.

[166] A. N. Netravali. *Digital pictures: representation, compression, and standards*. Plenum Press, New York, NY, USA, 1995.

[167] P. G. Neumann. Efficient error-limiting variable-length codes. *IRE Transactions on Information Theory*, 8(4), pp. 292–304, July 1962.

[168] K. Nguyen-Phi and H. Weinrichter. Bi-level image compression using adaptive tree model. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 458. IEEE Computer Society Press, Snowbird, UT, USA, March 1997.

[169] K. Nguyen-Phi and H. Weinrichter. Bi-level image compression using adaptive tree model. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 459. IEEE Computer Society Press, Snowbird, UT, USA, March 1997.

[170] D. Okkalides. Assessment of commercial compression algorithms, of the lossy DCT and lossless types, applied to diagnostic digital image files. *Computerized Medical Imaging and Graphics*, 22(1), pp. 25–30, January 1998.

[171] R. Pasco. *Source coding algorithms for fast data compression*. Ph.D. thesis, Department of Electrical Engineering, Stanford University, 1976.

[172] W. B. Pennebaker and J. L. Mitchell. *JPEG still image compression standard*. Van Nostrand Reinhold, New York, NY, USA, 1993.

[173] W. B. Pennebaker, J. L. Mitchell, G. G. Langdon, Jr., and R. B. Arps. An overview of the basic principles of the Q-coder adaptive binary arithmetic coder. *IBM Journal of Research and Development*, 32(6), pp. 717–726, November 1988.

[174] W. Philips and K. Denecker. A lossless version of the Hadamard transform. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 443–450. STW Technology Foundation, Mierlo, The Netherlands, November 1997.

[175] W. Philips and K. Denecker. A new embedded lossless/quasi-lossless image coder based on the Hadamard transform. In *Proceedings of the IEEE International Conference on Image Processing*, volume I, pp. 667–670. Santa Barabara, CA, USA, October 1997.

[176] W. Philips, K. Denecker, P. De Neve, and S. Van Assche. Very high quality compression of color and gray scale images: Techniques and applications. In *International Congress on Imaging Science, Electronic Imaging*, volume 2, pp. 259–263. Antwerp, Belgium, September 1998.

[177] W. Philips, K. Denecker, P. De Neve, and S. Van Assche. Lossless quantization of Hadamard transform coefficients. *IEEE Transactions on Image Processing*, 9(11), pp. 1995–1999, November 2000.

[178] W. Philips, S. Van Assche, D. De Rycke, and K. Denecker. State-of-the-art techniques for lossless compression of 3D medical image sets. *Computerized Medical Imaging and Graphics*, 25(2), pp. 173–185, March 2001.

[179] K. Pohlmann. *Principles of digital audio*. McGraw-Hill, New York, NY, USA, 1995.

[180] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 1992.

[181] C. Quenneville and J. Meunier. Image coding using finite state automata. *Optical Engineering*, 35(1), pp. 113–118, January 1996.

[182] M. Rabbani and P. W. Jones. *Digital image compression techniques*, volume TT7 of *Tutorial texts in optical engineering*. SPIE, 1991.

[183] M. Rabbani and P. W. Melnychuck. Conditioning contexts for the arithmetic coding of bit planes. *IEEE Transactions on Signal Processing*, 40(1), pp. 232–236, January 1992.

[184] R. F. Rice. Some practical universal noiseless coding techniques. Technical Report JPL-79-22, Jet Propulsion Laboratory, Pasadena, CA, USA, March 1979.

[185] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. World Scientific, Singapore, 1975.

[186] J. Rissanen. Universal coding, information, prediction, and estimation. *IEEE Transactions on Information Theory*, 30, pp. 629–636, July 1984.

[187] J. Rissanen. Stochastic complexity and modelling. *Annals of Statistics*, 14(3), pp. 1080–1100, September 1986.

[188] J. Rissanen. Fast universal coding with context models. *IEEE Transactions on Information Theory*, 45(4), pp. 1065–1071, May 1999.

[189] J. Rissanen and G. G. Langdon, Jr. Universal modeling and coding. *IEEE Transactions on Information Theory*, IT-27(1), pp. 12–23, January 1981.

[190] J. J. Rissanen. Generalized Kraft inequality and arithmetic coding. *IBM Journal of Research and Development*, 20, pp. 198–203, May 1976.

[191] J. J. Rissanen. A universal data compression system. *IEEE Transactions on Information Theory*, IT-29(5), pp. 656–664, September 1983.

[192] J. J. Rissanen and G. G. Langdon. Arithmetic coding. *IBM Journal of Research and Development*, 23(2), pp. 149–162, March 1979.

[193] J. J. Rissanen and K. M. Mohiuddin. A multiplication-free multialphabet arithmetic code. *IEEE Transactions on Communications*, 37, pp. 93–98, 1989.

[194] F. Rizzo, J. A. Storer, and B. Carpentieri. Improving single-pass adaptive VQ. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, volume 6, pp. 3169–3172. IEEE Computer Society Press, Phoenix, AR, USA, March 1999.

[195] J. A. Robinson. Efficient general-purpose image compression with binary tree predictive coding. *IEEE Transactions on Image Processing*, 6(4), pp. 601–608, April 1997.

[196] B. Rogge, I. Lemahieu, W. Philips, K. Denecker, P. De Neve, and S. Van Assche. Region of interest based progressive transmission of greyscale images across the Internet. In G. B. Beretta and R. Eschbach, eds., *Color Imaging: Device-Independent Color, Color Hard Copy and Graphic Arts IV*, volume 3648 of *Proceedings of SPIE*, pp. 365–372. SPIE, IS&T, San Jose, CA, USA, January 1999.

[197] B. Rogge, I. Lemahieu, W. Philips, K. Denecker, P. De Neve, and S. Van Assche. Region of interest based progressive transmission of greyscale images across the Internet. In W. Hahn, E. Walther-Klaus, and J. Knop, eds., *Euromedia*, pp. 28–32. Society for Computer Simulation International, Munchen, Germany, April 1999.

[198] B. Rogge, I. Lemahieu, W. Philips, K. Denecker, S. Van Assche, and P. De Neve. Region of interest based progressive transmission of greyscale images across the Internet. In G. Joubert and E. D'Hollander, eds., *High Performance Computing: Biomedical Applications and Parallel Architectures*, pp. 79–89. Communication & Cognition, Clausthal, Germany, January 1999.

[199] B. Rogge, I. Lemahieu, W. Philips, S. Van Assche, K. Denecker, and P. De Neve. ROI-based progressive transmission of medical images across the internet. In L. J. Lambrecht, ed., *European Symposium on Clinical Imaging and Networking*, p. 44. Antwerp, Belgium, April 1998.

[200] S. G. Romaniuk. Theoretical results for applying neural networks to lossless image compression. *Network: Computation in Neural Systems*, 5(4), pp. 583–597, November 1994.

[201] P. Roos and M. A. Viergever. Reversible 3-D decorrelation of medical images. *IEEE Transactions on Medical Imaging*, 12(3), pp. 413–420, September 1993.

[202] P. Roos, M. A. Viergever, M. C. A. van Dijke, and J. H. Peters. Reversible intraframe compression of medical images. *IEEE Transactions on Medical Imaging*, 7(4), pp. 328–336, December 1988.

[203] F. Rubin. Arithmetic stream coding using fixed precision registers. *IEEE Transactions on Information Theory*, IT-25(6), pp. 672–675, November 1979.

[204] H. Sagan. *Space-Filling Curves*. Springer Verlag, New York, NY, USA, 1994.

[205] A. Said and W. A. Pearlman. Reversible image compression via multiresolution representation and predictive coding. In B. G. Haskell and H. Hang, eds.,

Visual Commmunications and Image Processing, volume 2094, pp. 664–673. Cambridge, MA, USA, November 1993.

[206] A. Said and W. A. Pearlman. An image multiresolution representation for lossless and lossy compression. *IEEE Transactions on Image Processing*, 5(9), pp. 1303–1310, September 1996.

[207] A. Said and W. A. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(3), pp. 243–250, June 1996.

[208] H. Sakanashi, M. Iwata, and T. Higuchi. A lossless compression method for halftone images using evolvable hardware. In Y. Liu, K. Tanaka, M. Iwata, T. Higuchi, and M. Yasunaga, eds., *Evolvable Systems: From Biology to Hardware*, Lecture Notes in Computer Science, pp. 314–326. Springer Verlag, Tokyo, Japan, October 2001.

[209] D. Salomon. *Data compression*. Springer-Verlag, New York, NY, USA, 1997.

[210] H. Samet. The quadtree and related hierarchical structures. *ACM Computing Surveys*, 16(2), pp. 187–260, 1984.

[211] D. Santa-Cruz, T. Ebrahimi, J. Askelöf, M. Larsson, and C. A. Christopoulos. JPEG 2000 still image coding versus other standards. In *Applications of Digital Image Processing XXIII*, volume 4115 of *Proceedings of SPIE*, pp. 446–454. San Diega, CA, USA, July 2000.

[212] A. A. Sardinas and G. W. Patterson. A necessary and sufficient condition for the unique decomposition of coded messages. In *IRE Convention Record, Part 8*, pp. 104–108. 1953.

[213] D. Sarkar. Boolean function-based approach for encoding of binary images. *Pattern Recognition Letters*, 17(8), pp. 839–848, July 1996.

[214] S. A. Savari and R. G. Gallager. Generalized Tunstall codes for sources with memory. *IEEE Transactions on Information Theory*, 43(2), pp. 658–668, March 1997.

[215] K. Sayood. *Introduction to data compression*. Morgan Kaufmann Publishers, 1996.

[216] K. Sayood and K. Anderson. A differential lossless image compression scheme. *IEEE Transactions on Signal Processing*, 40(1), pp. 236–241, January 1992.

[217] M. Schindler. A fast renormalisation for arithmetic coding. In J. A. Storer and M. Cohn, eds., *Proceedings Data Compression Conference*, p. 572. IEEE Computer Society, Snowbird, UT, US, March 1998.

[218] B. Schneier. *Applied cryptography*. John Wiley & Sons, New York, 2nd edition, 1995.

[219] E. S. Schwartz. An optimum encoding with minimum longest code and total number of digits. *Information and Control*, 7(1), pp. 37–44, March 1964.

[220] P. J. Sementilli, A. Bilgin, J. H. Kasner, and M. W. Marcellin. Wavelet TCQ: submission to JPEG-2000. In A. G. Tescher, ed., *Applications of Digital Image Processing*, volume 3460 of *Proceedings of SPIE*, pp. 2–12. July 1998.

[221] C. E. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27, pp. 398–403, 1948.

[222] C. E. Shannon. Prediction and entropy of printed English. *Bell Systems Technical Journal*, 30, pp. 50–64, 1951.

[223] C. E. Shannon and W. Weaver. *The mathematical theory of communication*. University of Illinois Press, Urbana, IL, USA, 1949.

[224] J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Transactions on Signal Processing*, 41(12), pp. 3445–3462, December 1993.

[225] L. Shen and R. M. Rangayyan. A segmentation-based lossless image coding method for high-resolution medical image compression. *IEEE Transactions on Medical Imaging*, 16(3), pp. 301–307, June 1997.

[226] R. Shiavi. *Introduction to applied statistical signal analysis*. Academic Press, San Diego, CA, USA, 2nd edition, 1999.

[227] A. Sieminski. Fast decoding of the Huffman codes. *Information Processing Letters*, 26(5), pp. 237–241, January 1988.

[228] K. Simons. Value at risk — new approaches to risk management. *New England Economic Review*, 3, October 1996.

[229] A. Skodras, C. Christopoulos, and T. Ebrahimi. The JPEG 2000 still image compression standard. *IEEE Signal Processing Magazine*, 18(5), pp. 36–58, September 2001.

[230] M. J. Slattery and J. L. Mitchell. The Qx-coder. *IBM Journal of Research and Development*, 42(6), pp. 767–784, November 1998.

[231] M. Starkey and R. Bryant. Using ordered binary-decision diagrams for compressing images and image sequences. Technical Report CMU-CS-95-105, School of Computer Science — Carnegie Mellon University, Pittsburgh, PA, USA, January 1995.

[232] G. Starkweather. The future of electronic printing. In G. B. Beretta and R. Eschbach, eds., *Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts III*, volume 3300 of *Proc. of SPIE*, pp. 14–20. SPIE, IS&T, San Jose, CA, US, January 1998.

[233] J. A. Storer, ed. *Image and text compression*. Kluwer Academic Publishers, 1992.

[234] J. A. Storer. Lossless image compression by block matching. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 290–299. IEEE Computer Society Press, Snowbird, UT, USA, March 1996.

[235] J. A. Storer and T. G. Szymanski. Data compression via textual substitution. *Journal of the ACM*, 29(4), pp. 928–951, October 1982.

[236] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and Computational Harmonic Analysis*, 3(2), pp. 186–200, April 1996.

[237] M. Tanaka, H. Sakanashi, M. Mizoguchi, and T. Higuchi. Bi-level image coding for digital printing using genetic algorithm. *Electronics and Communications in Japan Part III—Fundamental Electronic Science*, 84(9), pp. 1–10, January 2001.

[238] D. Taubman. High performance scalable image compression with EBCOT. *IEEE Transactions on Image Processing*, 9(7), pp. 1158–1170, July 2000.

[239] W. J. Teahan. Probability estimation for PPM. University of Waikato, New Zealand, 1996. http://www.cs.waikato.ac.nz/~ wjt/papers/NZCSRSC.ps.gz.

[240] W. J. Teahan. *Modelling English Text*. Ph.D. thesis, University of Waikato, Department of Computer Science, Hamilton, New Zealand, May 1998.

[241] W. J. Teahan and J. G. Cleary. The entropy of English using PPM-based models. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 53–62. IEEE Computer Society Press, Snowbird, UT, USA, March 1996.

[242] A. Timmermann. Moments of Markov switching models. *Journal of Econometrics*, 96(1), pp. 75–111, May 2000.

[243] T. J. Tjalkens and F. M. J. Willems. Variable to fixed-length codes for markov sources. *IEEE Transactions on Information Theory*, IT-33(2), pp. 246–257, March 1987.

[244] S. Todd, G. G. Langdon, Jr., and J. Rissanen. Parameter reduction and context selection for compression of the gray-scale images. *IBM Journal of Research and Development*, 29(2), pp. 188–193, March 1985.

[245] T. D. Tran. The BinDCT: Fast muliplierless approximation of the DCT. *IEEE Signal Processing Letters*, 7(6), pp. 141–144, June 2000.

[246] B. P. Tunstall. *Synthesis of noiseless compression codes*. Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA, USA, 1968.

[247] R. Ulichney. *Digital Halftoning*. MIT Press, Cambridge, MA, USA, 1987.

[248] S. Van Assche. *Statistische modellering en verliesloze compressie*. Ph.D. thesis, Ghent University, Department of Electronics and Information Systems, Ghent, Belgium, 2001.

[249] S. Van Assche, K. Denecker, J. Crombez, and W. Philips. Value-at-Risk prediction using context modeling. In K. Klemm and P. Alstrøm, eds., *Applications of Physics in Financial Analysis - 2*, volume 24E, p. 14. European Physical Society, Liège, Belgium, July 2000.

[250] S. Van Assche, K. Denecker, J. Crombez, R. Vander Vennet, and I. Lemahieu. Value-at-Risk estimation using context modeling. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 537–548. STW Technology Foundation, Mierlo, The Netherlands, December 2000.

[251] S. Van Assche, K. Denecker, and P. De Neve. Evaluation of lossless compression techniques for high-resolution RGB and CMYK color images. *Journal of Electronic Imaging*, 8(4), pp. 415–421, October 1999.

[252] S. Van Assche, K. Denecker, W. Philips, and I. Lemahieu. Adaptation of the CCITT Group 4 fax standard to screened images. In *International Congress on Imaging Science, Electronic Imaging*, volume 2, pp. 294–297. Antwerp, Belgium, September 1998.

[253] S. Van Assche, K. Denecker, W. Philips, and I. Lemahieu. Lossless compression of three-dimensional medical images. In J.-P. Veen, ed., *Proceedings of the PRORISC IEEE Benelux Workshop on Circuits, Systems and Signal Processing*, pp. 549–553. STW Technology Foundation, Mierlo, The Netherlands, November 1998.

[254] S. Van Assche, K. Denecker, W. Philips, and I. Lemahieu. A comparison of lossless compression techniques for prepress color images. In K. Aizawa, ed., *Visual Communications and Image Processing*, volume 3653 of *Proceedings of SPIE*, pp. 1376–1383. SPIE, IS&T, San Jose, CA, USA, January 1999.

[255] S. Van Assche, K. Denecker, W. Philips, and I. Lemahieu. Lossless compression of color images using color decorrelation. In *Proceedings of the World Multiconference on Systems, Cybernetics and Informatics*, pp. 250–256. Orlando, FL, USA, August 1999.

[256] S. Van Assche, K. Denecker, W. Philips, and I. Lemahieu. State-of-the-art concerning lossless compression of three-dimensional medical images. In *Fifth Conference of the European Society for Engineering and Medicine*, pp. 573–574. Barcelona, Spain, May 1999.

[257] S. Van Assche, W. Philips, and I. Lemahieu. Lossless compression of prepress images using a novel colour decorrelation technique. *Pattern Recognition*, 32(3), pp. 435–441, March 1999.

[258] S. Verdú. Fifty years of Shannon theory. *IEEE Transactions on Information Theory*, 44(6), pp. 2057–2078, October 1998.

[259] M. Vetterli and J. Kovacevic. *Wavelets and Subband Coding*. Prentice Hall, Englewood Cliffs, NJ, USA, 1st edition, 1995.

[260] J. S. Vitter. Design and analysis of dynamic Huffman codes. *Journal of the ACM*, 34(4), pp. 825–845, October 1987.

[261] F. M. J. Volf, P. A. J. an Willems. The switching method: Elaborations. In *Proceedings of the 19th Symposium on Information Theory in the Benelux*, pp. 13–20. Veldhoven, NL, May 1998.

[262] P. A. J. Volf and F. M. J. Willems. Switching between two universal source coding algorithms. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 491–500. IEEE Computer Society Press, Snowbird, UT, USA, March 1998.

[263] G. K. Wallace. The JPEG still picture compression standard. *Communications of the ACM*, 34(4), pp. 30–44, apr 1991.

[264] M. J. Weinberger, J. J. Rissanen, and R. B. Arps. Applications of universal context modeling to lossless compression of gray-scale images. *IEEE Transactions on Image Processing*, 5(4), pp. 575–586, April 1996.

[265] M. J. Weinberger and G. Seroussi. Sequential prediction and ranking in universal context modeling and data compression. *IEEE Transactions on Information Theory*, 43(5), pp. 1697–1706, September 1997.

[266] M. J. Weinberger, G. Seroussi, and G. Sapiro. LOCO-I: A low complexity, context-based, lossless image compression algorithm. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 140–149. IEEE Computer Society Press, Snowbird, UT, USA, March 1996.

[267] M. J. Weinberger, G. Seroussi, and G. Sapiro. The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS. *IEEE Transactions on Image Processing*, 9(8), pp. 1309–1324, August 2000.

[268] T. A. Welch. A technique for high-performance data compression. *Computer*, 17(6), pp. 8–19, June 1984.

[269] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens. The context-tree weighting method: Basic properties. *IEEE Transactions on Information Theory*, 41(3), pp. 653–664, May 1995.

[270] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens. Context weighting for general finite-context sources. *IEEE Transactions on Information Theory*, 42(5), pp. 1514–1520, September 1996.

[271] R. N. Williams. Dynamic-history predictive compression. *Information Systems*, 13(1), pp. 129–140, 1988.

[272] R. N. Williams. *Adaptive data compression*. Kluwer Academic Publishers, 1991.

[273] I. H. Witten. *Managing gigabytes: compressing and indexing documents and images*. Van Nostrand Reinhold, 1994.

[274] I. H. Witten and T. C. Bell. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, IT-37(4), pp. 1085–1094, July 1991.

[275] I. H. Witten, T. C. Bell, H. Emberson, S. Inglis, and A. Moffat. Textual image compression: Two-stage lossy/lossless encoding of textual images. *Proceedings of the IEEE*, 82(6), pp. 878–888, June 1994.

[276] I. H. Witten, R. Neal, and J. G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6), pp. 520–540, June 1987.

[277] J. W. Woods and S. D. O'Neil. Subband coding of images. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-34(5), pp. 1278–1288, October 1986.

[278] X. L. Wu. Context selection and quantization for lossless image coding. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, p. 453. IEEE Computer Society Press, Snowbird, UT, USA, March 1995.

[279] X. L. Wu. An algorithmic study on lossless image compression. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 150–159. IEEE Computer Society Press, Snowbird, UT, USA, March 1996.

[280] X. L. Wu. Lossless compression of continuous-tone images via context selection, quantization, and modeling. *IEEE Transactions on Image Processing*, 6(5), pp. 656–664, May 1997.

[281] X. L. Wu and P. Bao. L_∞ constrained high-fidelity image compression via adaptive context modeling. *IEEE Transactions on Image Processing*, 9(4), pp. 536–542, April 2000.

[282] X. L. Wu, W.-K. Choi, and N. Memon. Lossless interframe image compression via context modeling. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 378–387. IEEE Computer Society Press, Snowbird, UT, USA, March 1998.

[283] X. L. Wu and N. Memon. Context-based, adaptive, lossless image coding. *IEEE Transactions on Communications*, 45(4), pp. 437–444, April 1997.

[284] X. L. Wu and N. Memon. Context-based lossless interband compression — extending CALIC. *IEEE Transactions on Image Processing*, 9(6), pp. 994–1001, June 2000.

[285] G. Wyszecki and W. S. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley, 1982.

[286] A. Zandi, J. D. Allen, E. L. Schwartz, and M. Boliek. CREW: Compression with reversible embedded wavelets. In J. A. Storer and M. Cohn, eds., *Proceedings of the IEEE Data Compression Conference*, pp. 212–212. IEEE Computer Society Press, Snowbird, UT, USA, March 1995.

[287] J. Ziv. Variable-to-fixed length codes are better than fixed-to-variable length codes for Markov sources. *IEEE Transactions on Information Theory*, 36(4), pp. 861–863, July 1990.

[288] J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3), pp. 337–343, May 1977.

[289] J. Ziv and A. Lempel. Compression of individual sequences via variable rate coding. *IEEE Transactions on Information Theory*, IT-24(5), pp. 530–536, September 1978.