University of Ljubljana
Faculty of Computer and Information Science

Domen Rački

Object detection with constellations of keypoints

MASTERS THESIS

THE 2ND CYCLE MASTERS STUDY PROGRAMME
COMPUTER AND INFORMATION SCIENCE

Supervisor: doc. dr. Matej Kristan
Co-supervisor: Univ.-Prof. Dipl.-Ing. Dr. Thomas Pock

Ljubljana, 2015

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Domen Rački

Detekcija objektov s konstelacijami značilk

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: doc. dr. Matej Kristan

Somentor: Univ.-Prof. Dipl.-Ing. Dr. Thomas Pock

Ljubljana, 2015

Declaration of Masters Thesis authorship

I, the undersigned Domen Rački am the author of the Master Thesis entitled:

Object detection with constellations of keypoints

With my signature, I declare that:

- the submitted Thesis is my own unaided work under the supervision of doc. dr. Matej Kristan and co-supervision of Univ.-Prof. Dipl.-Ing. Dr. Thomas Pock,
- all electronic forms of the Masters Thesis, title (Slovenian, English), abstract (Slovenian, English) and keywords (Slovenian, English) are identical to the printed form of the Masters Thesis,
- I agree with the publication of the electronic form of the Masters Thesis in the collection "Dela FRI".

In Ljubljana, 10. April 2015 Author's signature:

Acknowledgments

The thesis, in a broader aspect, was affected by different phases and different timelines. By those who are and those who where. My sincere gratitude to all.

My sincere gratitude to my supervisor doc. dr. Matej Kristan for his patience, motivation, enthusiasm, and for his willingness to share his immense knowledge. Every academic and non-academic advice in my endeavor was greatly appreciated.

My sincere gratitude to my co-supervisor Univ.-Prof. Dipl.-Ing. Dr. Thomas Pock for taking over my supervision during my exchange study period at the Graz University of Technology, Austria. Danke für alles.

My sincere gratitude to the Faculty of Computer and Information Science and the Graz University of Technology for giving me the opportunity to participate in the double-degree programme. This opportunity was the first flutter of butterfly wings.

My sincere gratitude to doc. dr. Danijel Skočaj and the whole ViCoS laboratory for all the new and invaluable theoretical and practical knowledge, all the support and all the discussions. And all the coffee!

My sincere gratitude to my parents, to whom I dedicate this thesis to. Thank you Majda for showing me that no challenge is too big. Thank you Jože for showing me that no path is too long. Thank you for your unconditional and everlasting love and support. Thank you for always believing in me.

My sincere gratitude to my whole family for all their love and support. Thank you Aleksej and Neli for all the playtimes. Learn well! Thank you Jelka for all the baked goods. Žiga and Jana, thank you for the cake!

Last, but no least, my sincere gratitude to all my friends and to Klavdija. Thank you for all the happy times and for normalizing my mindset every time the smallest detail threatened to capture my entire focus.

Domen Rački, 2015

To Majda and Jože.

"Natürlicher Verstand kann fast jeden Grad von Bildung ersetzen, aber keine Bildung den natürlichen Verstand."

— Arthur Schopenhauer

Contents

Povzetek

Metode za detekcijo objektov osnovane na značilnicah se za določitev lokacije specifičnega objekta v testni sliki zanašajo na diskriminativno naravo značilnic. Nediskriminativne značilnice v množici detektiranih značilnic se izloča z uporabo podobnostnega pragu. To pomeni da se detektirano značilnico zavrže, če je ta podobna več kot eni značilnici v modelu. V primerih detekcije objektov s ponovljivimi se vzorci se podobnosti prag izkaže kot neučinkovit, saj obravnava večino detektiranih značilnic kot nediskriminativne, t.j., podobnih več kot eni značilnici v modelu. V kontekstu učenja z enim primerom v magistrski nalogi predlagamo konstelacijski model kot dodatek k osnovnim metodam za detekcijo objektov, osnovanih na zančilnicah. Cilj je uporabiti ohranjeno geometrijo med značilnicami kot filter za nediskriminativne značilnice in posledično eliminirati potrebo po podobnostnem pragu. Delovanje predlaganega konstelacijskega modela z empirično in numerično varianco značilnic primerjamo z osnovnim modelom osnovanim na značilnicah. Model evaluiramo na zahtevni bazi katera se sestoji iz logotipov v realnih okoljih. Ugotovimo da je najboljša različica konstelacijskega modela tista z empirično varianco značilnic, saj slednja značilno zmanjša število nediskriminativnih značilnic brez značilnega poslabšanja delovanja algoritma za detekcijo objektov.

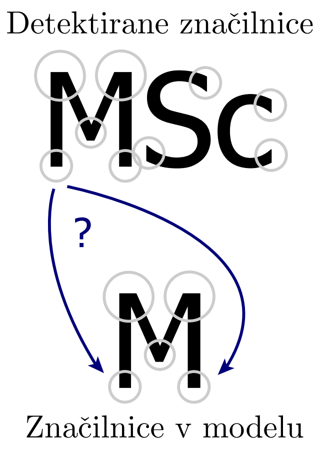
Ključne besede

učenje z enim primerom, značilnice, geometrija, varianca, konstelacije, detekcija objektov, SIFT, GHT, MLESAC, MND

Razširjeni povzetek

Detekcija objektov je proces identifikacije specifičnih objektov v digitalnih slikah, t.j., digitalnih videjih. Metode za detekcijo objektov tipično uporabljajo tako imenovane silkovne "značilnice" ter učne algoritme. Slikovne značilnice so tipično specifične strukture pridobljene iz visoko kontrastnih slikovnih regij, kot so točke in robovi, npr. [1], lahko pa so tudi rezultat splošne operacije v okolici visoko kontrastnih regij, npr. [2]. Pred izvedbo dejanske detekcije specifičnega objekta je potrebno konstruirati abstraktno reprezentacijo objekta, tako imenovani model objekta. Model je konstruiran z uporabo slikovnih značilnic pridobljenih iz slike ki prikazuje objekt zanimanja. V računalniškem vidu *učenje z enim primerom* [3] označuje problem izgradnje vizualnega model iz ene same učne slike. Glede na to, da je na voljo zgolj en učni primer je glavni poudarek na izgradnji robustnega vizualnega modela, katerega se nato uporabi v procesu detekcije objekta. Prednosti učenja vizualnega modela iz množice slik so pri učenju z enim primerom izgubljene, kar pomeni, da je potreben drugačen pristop za izgradnjo vizualnega modela.

Sam proces detekcije objektov z uporabo konstruiranega modela se sestoji iz večih korakov. Vsaki značilnici, pridobljeni iz poljubne testne slike ki vsebuje ali ne vsebuje objekt zanimanja, se v abstratknem modelu objekta poišče najbolj podobno značilnico, t.j., najbližji sosed. Iz množice ujemajočih se značilnic se z metodo za robustno ocenjevanje točkovnih korespondenc poišče podmnožica najbolje se ujemajočih značilnic, katere slikajo model objekta na specifično lokacijo v testni sliki. Sledijo lahko dodatni koraki



Slika 1: Ilustracija primera kjer detektirani značilnici, označeni s sivim krogcem, zaradi nediskriminativne narave značilnice ni mogoče prirediti pravilno značilnico v modelu, prav tako označeno s sivim krogcem.

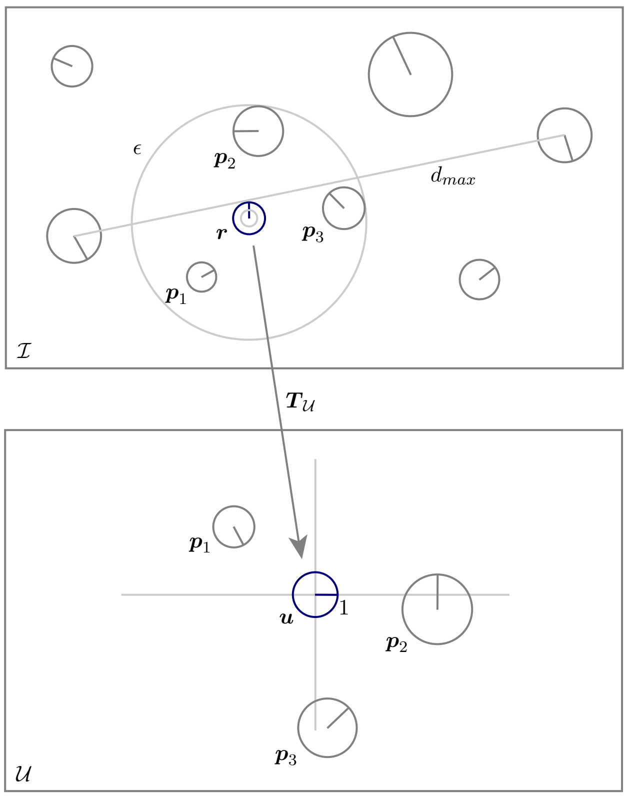
odstranjevanja šumnih značilnic in verifikacije detekcije. Detekcija objektov ima širok spekter aplikabilnosti tako v specializiranih kot vsakdanjih izzivih računalniškega vida. Tovrstne aplikacije segajo od sistemov za poizvedovanje po slikovnih zbirkah [4, 5], nadzornih sistemov [6], avtomatiziranih "poberi-in-odloži" sistemov [7] do prepoznavanja vsakdanjih objektov [8, 9, 10].

Kot je ilustrirano v Sliki 1 pri iskanju najbolj podobnih značilnic z metodo najbližjih sosedov ni mogoče povsem preprečiti šumne povezave, kjer se nediskriminativna detektirana značilnica poveže z naključno značilnico v modelu. Za preprečevanje tovrstnih šumnih povezav se v [11] predlaga uporaba praga podobnosti θ. Potencialna povezava med dvema značilnicama $\boldsymbol{k}_i \rightarrow \boldsymbol{m}_i$ se zavrže, če je prag podobnosti med detektirano značilnico $\boldsymbol{k}_i$ in dvema značilnicama modela $\boldsymbol{m}_i$ in $\boldsymbol{m}_j$ večji od θ, t.j., $\frac{s_1}{s_2} > \theta$, kjer je $s_1 = \boldsymbol{k}_i \sim \boldsymbol{m}_i$ in $s_2 = \boldsymbol{k}_i \sim \boldsymbol{m}_j$. Čeprav predlagani pristop prepreči povezave z nediskriminativnimi značilnicami, hkrati predstavlja potencialni problem. Recimo da je objekt ki ga želimo detektirati črka "M", kot prikazano v Sliki 1.1. Značilnice ki predstavljajo levi spodnji del in desni spodnji del črke "M" so identitčne. Pri poizkusu detekcije objekta, črke M, na predloženi testni sliki se zgodi naslednje. Ko poizkusimo poiskati povezave med detektiranimi značilnicami in značilnicami v modelu, predlagani prag podobnosti zavrže potencialno uporabne povezave, saj je značilnica, detektirana na levem spodnjem delu črke M v predloženi sliki zelo podobna dvema značilnicama v modelu ki predstavljata levi spodnji in desni spodnji del našega objekta, črke M. V kolikor ignoriramo prag podobnosti in iščemo povezave med značilnicami zgolj z uporabo najbližjih sosedov, pa v danem primeru ni jasno katera od dveh značilnic v modelu je najbolj podobna značilnici, detektirani na levem spodnjem delu črke M v predloženi sliki. Z uporabo praga podobnosti [11] torej res zmanjšamo število šumnih povezav, katere generirajo napačne hipoteze o lokaciji objekta v slikah, a hkrati izgubimo koristne informacije o lokaciji objekta. Omenjena pomanjkljivost je posebej opazna v primerih detekcije objektov s ponovljivimi se vzorci, kot so npr. večkratne pojavitve enake črke v logotipih.

Metode za detekcijo objektov v računalniškem vidu uporabljajo vrsto različnih pristopov, kot je dreseče okno [12], ujemanje predloge [13], segmentacija slike in analiza regij [14], model vreče besed [15] in različne metode, osnovane na slikovnih značilnicah [11, 16, 2, 17]. Glavna slabost pristopa z drsečim oknom [12] je veliko število okenj različnih skal ki jih je potrebno evaluirati, ter potreba po ustrezni množici testnih primerov na podlagi katerih se uči klasifikacijski algoritem. Princip ujemanje predloge [13] temelji na iskanju regije v sliki ki sovpada z naučeno predlogo. Tovrstni pristopi ne neaslavljajo deformacij objektov ter niso robustni na delna zakrivanja objektov. Pristopi s predhodno segmentacijo slike in naknadno analizo regij [14] so zelo odvisni od delovanja segmentacijskih metod, ki pa se izkažejo za problematične pri nasičenih regijah v sliki in pri naslavljanju različnih skal objekta. Pristop z modelom vreče besed [15] sicer bazira na značilnicah, toda ne uporablja ohranjene geometrije med značilnicami, kot dodatni filter šumnih značilnic, ter za delovanje potrebuje naučeni klasifikacijski algoritem. Pristopi osnovani na slikovnih značilnicah uporabljajo različne algoritme za detekcijo značilnic [11, 16, 2, 17]. Eden takšnih je SIFT [11], katerega značilnice so invariantne na skalo in rotacijo. Ker se slednji izkaže kot eden najbolj robustnih algoritmov za detekcijo značilnic v [18], ga tudi sami uporabimo v magistrski nalogi. Scale-invariant feature transform [11] (SIFT) je algoritem za detekcijo značilnic ki se uporablja v sodobnih pristopih detekcije objektov. Algoritem detektira in opiše lokalne značilnice v procesu detekcije ekstremov v prostoru skal, lokalizacija značilnih točk, prirejanje orientacije in prirejanje opisnika regije vsaki značilnici. SIFT značilnica se tako sestoji iz detektirane značilne točke ki je predstavljena z lokacijo, skalo in orientacijo, ter prirejenega opisnika regije. Od objave SIFT so bili objavljeni številni drugi algoritmi ki bazirajo na SIFT z različnimi modifikacijami [19, 20, 21, 18, 22]. Učenje z enim primerom v kontekstu kategorij objektov lahko najdemo v [3], kjer avtorji predstavijo pristop k učenju vizualnih modelov kategorij objektov z uporabo verjetnostnih modelov. Pristop detekcije objektov z uporabo ohranjenih prostorskih relacij med značilnicami lahko najdemo v [23, 24], kjer avtorji enkodirajo in v testnih slikah iščejo trojčke značilnic, t.j., povezave med tremi značilnicami. Celoten proces zahteva več različnih učnih slik, problem deformacije objektov pa je naslovljen z deformacijo učnih oz. testnih slik saj na takšen način avtorji simulirajo deformacije med prostorskimi relacijami značilnic.

V magistrski nalogi generaliziramo tehniko enkodiranja na poljubno število sosednjih značilnic, deformacije med prostorskimi relacijami pa modeliramo z dvema pristopoma in tako eleiminiramo potrebo bo množici učnih slik. Naš pristop torej zahteva zgolj eno planarno sliko za učenje objekta, t.j., učenje z enim primerom. V nadaljevanju povzamemo postopek gradnje konstelacijskega model ter postopek detekcije objekta s konstelacijami.

Na podlagi značilnic pridobljenih iz slike ki prikazuje poljubni objekt konstruiramo abstraktno reprezentacijo objekta, t.j., model $\mathcal{M}$. Za vsako značilnico enkodiramo prostorske relacije med sosednjimi značilnicami, kot je prikazano v Sliki 2, s čim pridobimo konstelacije značilnic. V vsaki konstelaciji se modelirajo pričakovane variance enkodiranih značilnic, kot je prikazano v Sliki 3, s čimer pridobimo deformabilne konstelacije katere lahko naslovijo deformacije lokalinih regij. Variance značilnic se modelirajo na dva

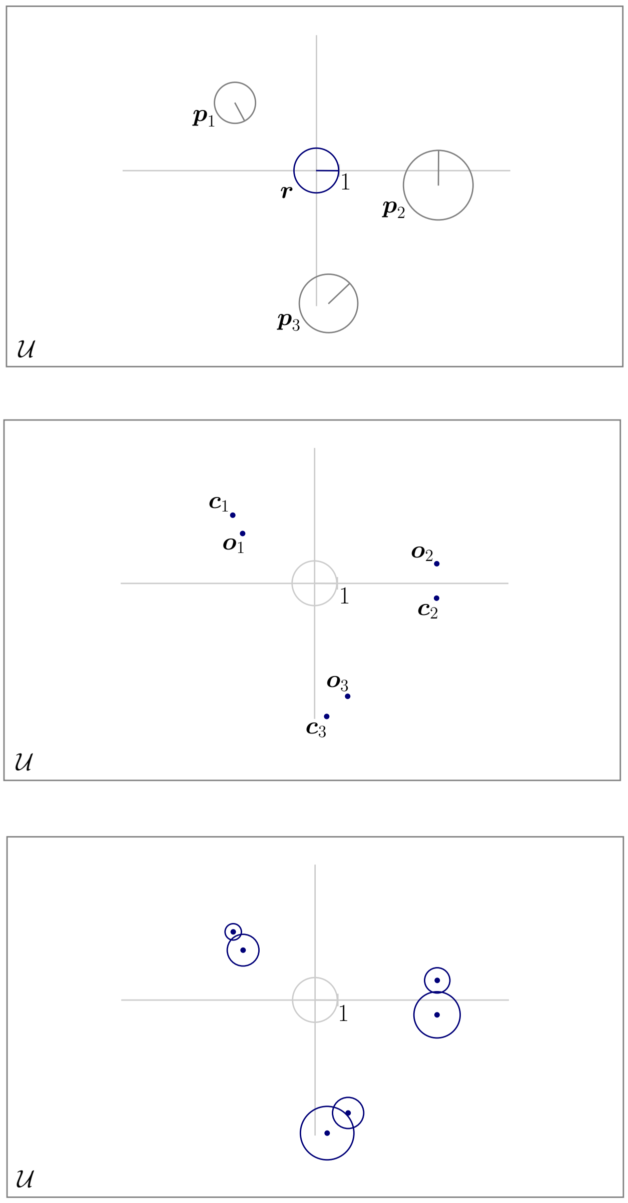


Slika 2: Značilnice v kooridantnem sistemu slike $\mathcal{I}$, katere ležijo znotraj ϵ regije okoli korenske značilnice $\boldsymbol{r}$, se smatra kot sosednje značilnice. Transformacija $\boldsymbol{T}_{\mathcal{U}} : \boldsymbol{r} \rightarrow \boldsymbol{u}$ preslika vse sosednje značilnice v normalizirani enotski prostor $\mathcal{U}$ in posledično enkodira lokalno soseščino korenske značilnice.

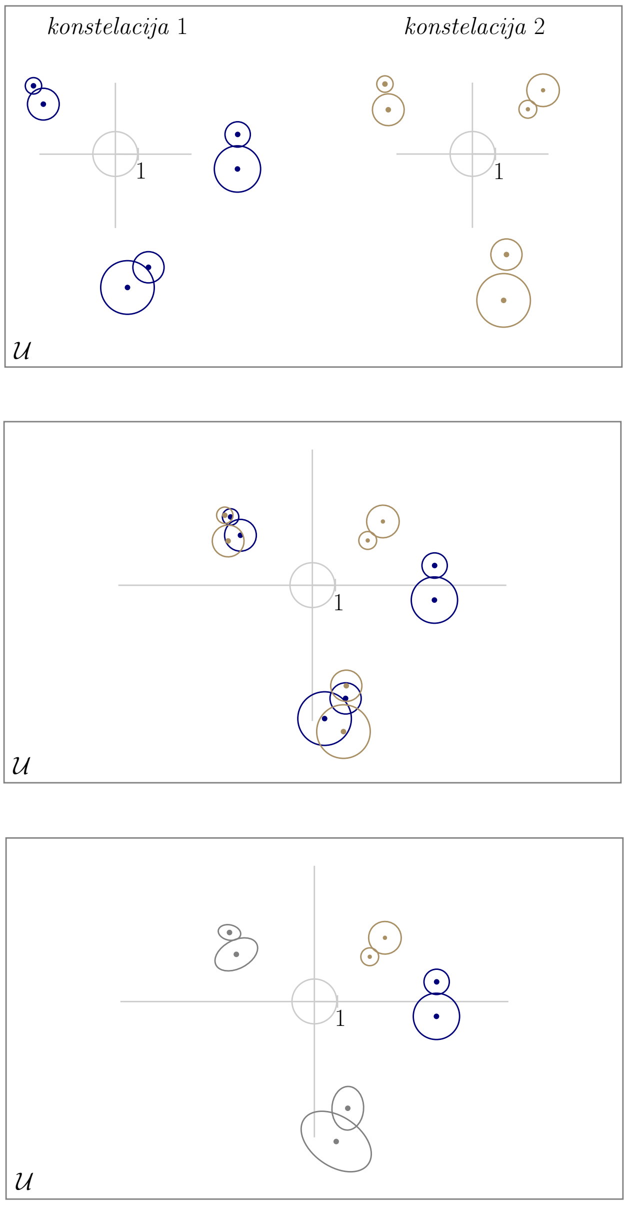
načina in sicer empirično ter numerično. Vse konstelacije so enkodirane v skupnem prostoru, tako imenovanem *enotskem prostoru*. Podobne variance značilnic v enotskem prostoru združimo skupaj in s tem zmanjšamo skupno število enkodiranih varianc, kot je prikazano v Sliki 4. Za vsak objekt se poleg enkodiranih prostorskih relacij med značilnicami v model shranijo še središčna točka, okvir objekta, ter intenzitetna predloga objekta. Končna reprezentacija modela je ilustrirana v Sliki 5.

Postopek detekcije objekta s konstelacijskim modelom je sledeč. Značilnicam, pridobljenim iz poljubne testne slike ki vsebuje ali ne vsebuje objekt zanimanja, se poišče najbolj podobne značilnice v modelu. Za vsako značilnico se izvede filtriranje s pomočjo enkodiranih konstelacij z namenom izločitve značilnic ki se ne nahajajo na objektu zanimanja. Na podlagi preostalih značilnic se določijo potencialne lokacije objekta, t.j., lokacijske hipoteze. Za vsako lokacijsko hipotezo se oceni lokacija objekta v testni sliki s pomočjo ujemajočih se značilnic ter shranjenega okvirja. Ocenjena lokacija se izboljša s pomočjo iterativnega ocenjevanja okvirja. Kot zadnji korak se izvede verifikacija detekcije z uporabno normalizirane navzkrižne korelacije med ocenjeno lokacijo objekta v testni sliki ter intenzitetno predlogo objekta. Če izračunana vrednost presega prag τ_{ncc}, detekcijo proglasimo za uspešno.

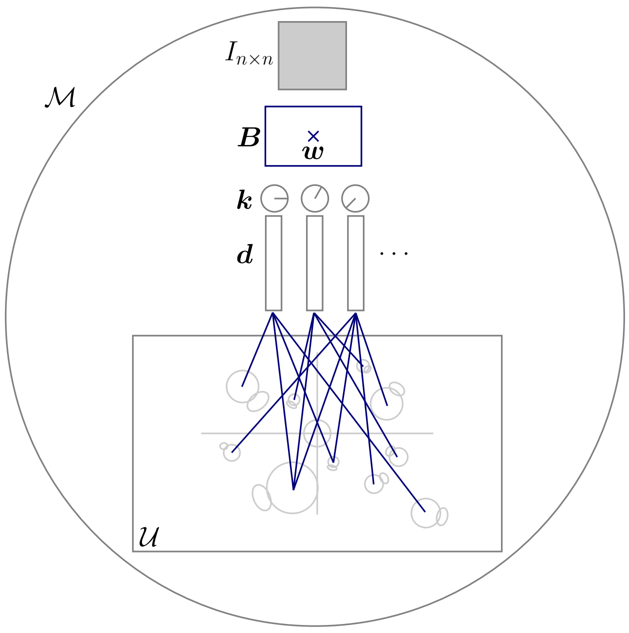
Cilj predlaganega konstelacijskega modela je, da za dani objekt filtrira značilnice pridobljene iz testne slike katere po gemoetrijskem aspektu ne sovpadajo z značilnicami enkodiranimi v modelu, in s tem reducira število lokacijskih hipotez ki jih je potrebno preveriti. Eksperimentalno evaluiramo šest različnih modelov: (i) osnovni model $\mathcal{B}$, (ii) osnovni model z uporabo podobnostnega praga $\mathcal{B}_\tau$, (iii) konstelacijski model z empirično varianco $\mathcal{C}^E$, (iv) konstelacijski model z empirično varianco in podobnostnim pragom $\mathcal{C}^E_\tau$, (v) konstelacijski model z numerično varianco $\mathcal{C}^N$, (vi) konstelacijski model z numerično varianco in podobnostnim pragom $\mathcal{C}^N_\tau$. Povzetek modelov je dan v Tabeli 1. Graf 6 prikazuje celotno število generiranih hipotez H_{total} in število hipotez ki presegajo prag za izločitev šumnih lokacijskih hipotez $H_{\tau_{GHT}}$, medtem ko Graf 7 prikazuje povprečno število generiranih hipotez



Slika 3: V enotskem prostoru je vsaka značilnica predstavljena z dvema točkama, t.j., središčna točka $\boldsymbol{c}$ in kotna točka $\boldsymbol{o}$. Na obeh točki značilnice se "vpne" pričakovana varianca, s čim se tvori deformabilna konstelacija značilnic.



Slika 4: Določene variance v enotskem prostoru se prekrivajo, torej so iz geometričnega vidika zelo podobne. Podobne variance značilnic združimo skupaj in s tem zmanjšamo skupno število enkodiranih varianc.



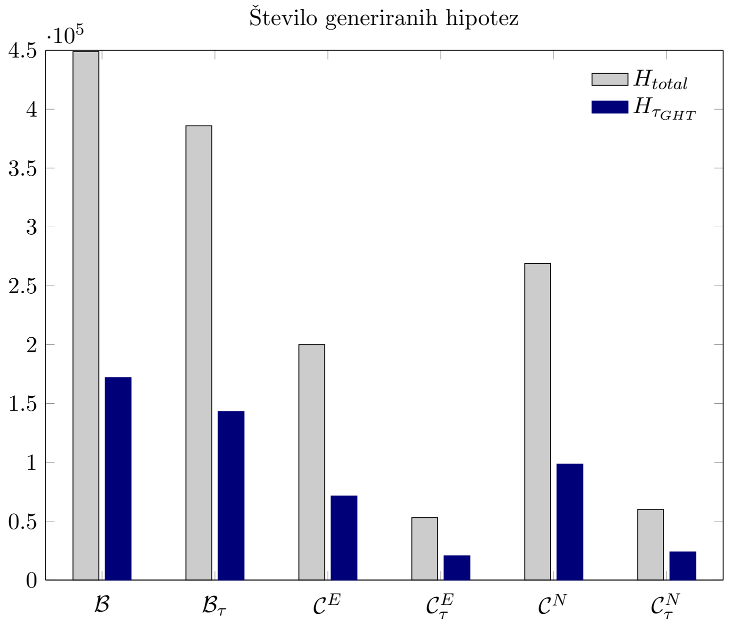
Slika 5: Reprezentacija modela $\mathcal{M}$. Vsaka značilnica $\boldsymbol{k}_i$ je predstavljena z ustreznim deskriptorjem $\boldsymbol{d}_i$, kateri kaže na eno ali več varianc v enotskem prostoru, s čim označuje konstelacijo okoli značilnice.

Tabela 1: Testirani modeli.

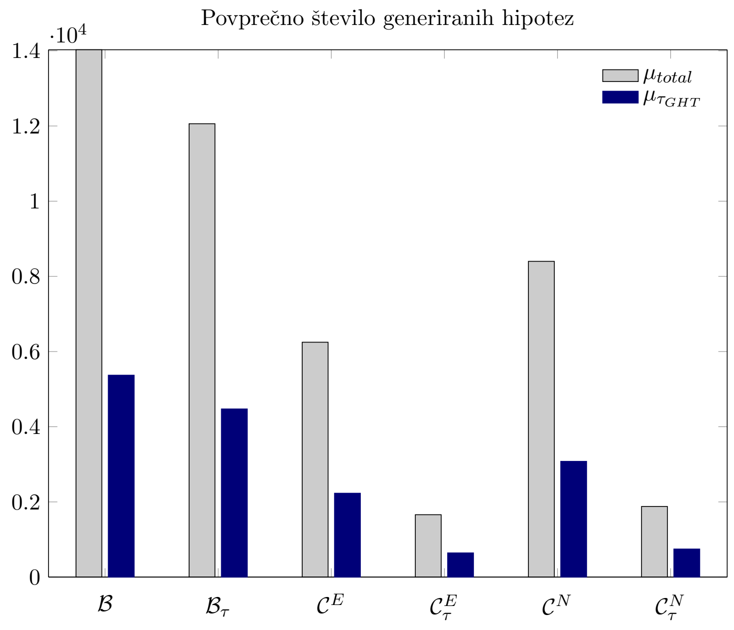
Notacija modela	Prag podobnosti	Empirična varianca	Numerična varianca
$\mathcal{B}$	$\times$	$\times$	$\times$
$\mathcal{B}_\tau$	✓	$\times$	$\times$
$\mathcal{C}^E$	$\times$	✓	$\times$
$\mathcal{C}^E_\tau$	✓	✓	$\times$
$\mathcal{C}^N$	$\times$	$\times$	✓
$\mathcal{C}^N_\tau$	✓	$\times$	✓

μ_{total} in $\mu_{\tau_{GHT}}$. Predlagani konstelacijski model v obeh primerih značilno zmanjša število lokacijskih hipotez ki jih je potrebno preveriti. Graf 8 prikazuje krivuljo preciznost-preklic. V splošnem ni razvidna značilna razlika med performancami modelov kar implicira da konstelacije ne poslapšajo delovanja osnovnega modela. Graf 9 prikazuje F-mero v odvisnosti od τ_{ncc}. Tudi tu ni razvidnih značilnih razlik med testiranimi modeli, t.j., graf nakazuje podoben trend vseh testiranih modelov.

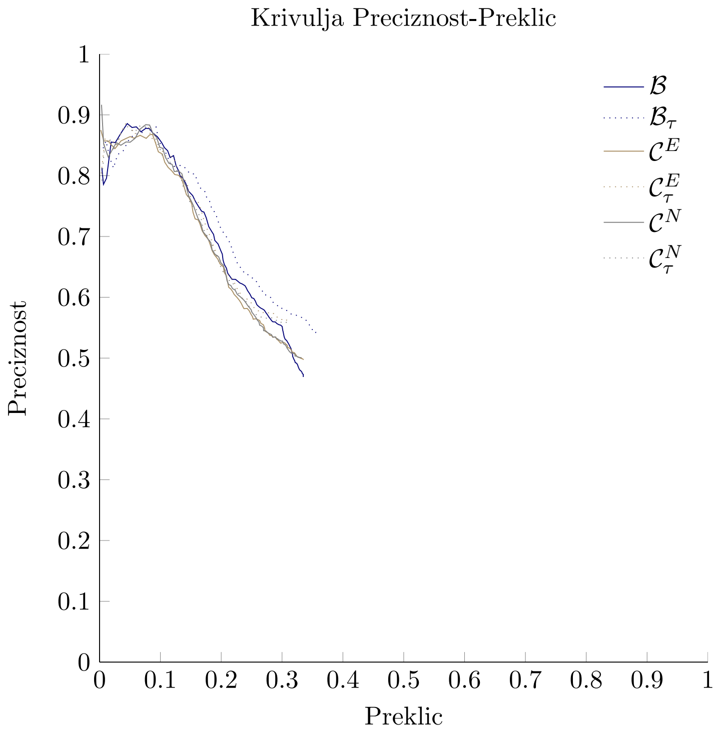
V magistrski nalogi predlagamo konstelacijski model kot dodatek za alogoritem za detekcijo značilnic in s tem nadomestimo potrebno po podobnostnem pragu, saj se slednji v določenih primerih izkaže za uporabnega. Predlagani konstelacijski model filtrira šumne značilnice in posledično zmanjša število lokacijskih hipotez, pri čemer ne vpliva značilno na delovanje algoritma. Filtriranje bazira izključno na geometriji saj se kot filter uporabljajo prostorske relacije med značilnicami in ne podobnosti med značilnicami. Dodatna prednost predlaganega model je tudi v tem da ga je moč uporabiti kot dodatek za poljubni alogoritem za detekcijo objektov, osnovanim na značilnicah, saj model izkorišča enotno predstavitev značilnice velike večine algoritmov za detekcijo značilnic, konstelacije pa se enkodirajo in uporabljajo nad detektiranimi značilnic.



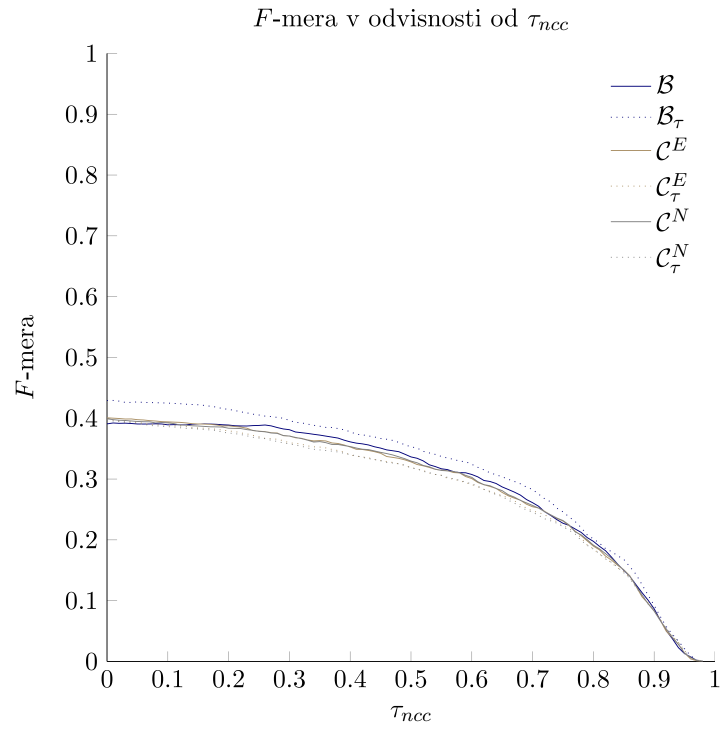
Slika 6: Celotno število generiranih hipotez H_{total} ter število generiranih hipotez ki presegajo prag za izločitev šumnih lokacijskih hipotez $H_{\tau_{GHT}}$. V obeh primerih predlagani konstelacijski model značilno zmanjša število generiranih hipotez, tako brez kot z uporabo podobnostnega pragu.



Slika 7: Povprečno število vseh generiranih hipotez μ_{total} ter povprečno število generiranih hipotez ki presegajo prag za izločitev šumnih lokacijskih hipotez $\mu_{\tau_{GHT}}$. V obeh primerih predlagani konstelacijski model značilno zmanjša povprečno število generiranih hipotez, tako brez kot z uporabo podobnostnega pragu.



Slika 8: Evaluacija testiranih modelov glede na krivuljo preciznost-preklic. Performance modelov se ne razlikujejo signifikantno, kar implicira da vpliv predlaganega konstelacijskega modela na zmogljivost algoritma za detekcijo ni značilen. Torej bistevno ne poslabša ali izboljša delovanje.



Slika 9: F-mera v odvisnosti od pragovne vrednosti normalizirane navzkrižne korelacije τ_{ncc}. F-mera testiranih modelv se ne razlikuje bistveno, kar implicira da vpliv predlaganega konstelacijskega modela na zmogljivost algoritma za detekcijo ni značilen. Torej bistevno ne poslabša ali izboljša delovanje.

Abstract

Feature-based object detection methods rely on the discriminative nature of features in order to accurately determine the location of a specific object in a test image. From a set of detected features, non-discriminative features are filtered out by means of a similarity threshold, meaning that if a features is very similar to more than one model feature, it is considered to be non-discriminative. However, in cases where an object consists of repeating patterns the similarity threshold proves inefficient since it considers the majority of detected features to be similar to more than one model feature, i.e., non-discriminative. In the context of one-shot learning we propose a constellation model for enhancing basic feature-based object detection methods, with the aim in utilizing the preserved geometry between features to filter out noisy feature matches. This eliminates the need for the similarity threshold. We evaluate the proposed constellation model whit empirically and numerically modelled feature variance and compare it to a baseline feature model. Model evaluation is performed on a challenging real-world dataset, consisting of logotypes in real-world scenarios. We find that the best variation of the constellation model is the model with empirically determined feature variance, which significantly reduces the number of mismatched features, without significantly affecting detection performance.

Keywords

one-shot learning, keypoints, geometry, variance, constellation, object detection, SIFT, GHT, MLESAC, MND

Chapter 1

Introduction

Object detection is the process of identifying instances of specific objects in digital images or series of digital images, i.e., digital videos. Object detection methods typically utilize extracted image ”features” and learning algorithms. Image features are usually specific structures, obtained from high-contrast regions in images, such as points or edges, e.g. [1], but may also be the result of a general neighbourhood operation around a high-contrast region, e.g. [2]. Before a specific object can be detected, an abstract representation of the object, called the model, is constructed utilizing extracted features from an image depicting the object of interest. In computer vision, *one-shot learning* [3] denotes the problem of constructing a visual model from a single training image. Given that a single training image is available, the main focus is on the construction of a robust visual model which in turn is utilized for the object detection task. The benefits of learning a visual model from a set of training images are lost, meaning that a different approach for the construction of a visual model is required.

The object detection task itself, utilizing the constructed model, consists of several steps. Features are extracted from an arbitrary test image containing, or not containing, the object of interest, and are matched to features in the abstract model representation of the object. Feature matching is typically done in a nearest-neighbour manner, meaning that every extracted feature

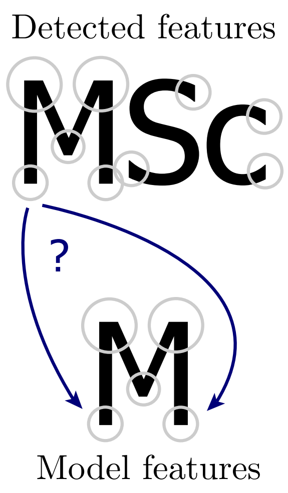


Figure 1.1: Illustration of a case where a detected feature, represented by a gray circle, cannot be correctly matched to a model feature, also represented by a gray circle, due to its non-discriminative nature.

is matched to the most similar feature in the model. A method for robustly estimating point correspondences is applied to the set of matched features in order to find a subset of best matching features, which map the object model to a specific location in the test image. Additional post detection-verification and steps towards reducing noisy feature matches may also be applied. Object detection has a wide range of applications in specialized as well as everyday computer vision tasks. The applications range from image retrieval systems [4, 5], security and surveillance systems [6], automated pick-and-place systems [7] to recognition of everyday object [8, 9, 10].

As illustrated in Figure 1.1, using only the nearest-neighbour matching technique, noisy mismatches, where a potentially non-discriminative detected feature is matched to a random model feature cannot be avoided. In order to address noisy mismatches and reject non-discriminative features, a similarity threshold θ is proposed in [11]. A potential feature match $\boldsymbol{k}_i \rightarrow \boldsymbol{m}_i$ is rejected, if the similarity threshold between the detected feature $\boldsymbol{k}_i$ and two model features $\boldsymbol{m}_i$ and $\boldsymbol{m}_j$ is higher than θ, i.e., $\frac{s_1}{s_2} > \theta$, where $s_1 = \boldsymbol{k}_i \sim \boldsymbol{m}_i$ and $s_2 = \boldsymbol{k}_i \sim \boldsymbol{m}_j$. This does in fact reject non-discriminative features but also presents a potential problem. Consider that our object is the letter "M", as illustrated in Figure 1.1. The features representing the bottom-left and the bottom-right "leg" of the letter M are identical. If we now present a test image on which we would like to detect our object, the letter M, the following happens. When we try to match detected features to model features, the proposed threshold rejects potentially useful matches, since the feature detected on the bottom-left "leg" of the letter M in the presented image is very similar to two model features, representing the bottom-left and the bottom-right "leg" of the letter M object. If, however, we ignore the similarity threshold and match using only the nearest-neighbour technique, we see that it is not clear which of the two features is the better match for the feature detected on the bottom-left "leg" of the letter M in the proposed image.

Another approach towards addressing noisy matches would be by using the nearest-neighbour technique to group together similar features within the model. However, in the proposed case, this also proves inefficient. If we would group together the two features representing the bottom-left and bottom-right "leg" of the letter M object in the model, we eliminate the problem of matching the two detected "leg" features to the model, since both get matched to the "same" grouped model feature, but this causes confusion when it comes to point-to-point correspondences. From a geometrical perspective the two features lie in different locations. If both locations get matched to the same model feature, both are very likely to get rejected by a method for robustly estimating point correspondences, since the provided location information for estimating the object location proves to be noisy.

In short, by using the similarity threshold [11] we reduce the noisy matches, which affect detection performance, but in turn loose valuable information about the object location. If the similarity threshold is ignored, noisy matches which interfere with the object detection process are not addressed. Grouping similar model features eliminates the problem of matching detected features, but in turn introduces noise when it comes to estimating the object location, since multiple detected features can ”point” to the same grouped feature encoded in the model. These shortcomings are noticeable when detecting objects containing repeating patterns such as multiple instances of the same letter in logotypes.

In the master’s thesis we address the problem of noisy feature matches by looking at constellations of features. Since feature-based detection methods preserve spatial relations between detected features in images, even under image deformations and view point changes, they are ideal for building constellations of features, with the aim in utilizing the preserved geometry to aid the detection task, by filtering out features not corresponding to a specific constellation. Given that in real-world scenarios objects will most certainly be subjected to different perspective deformations, we model the expected deformations of constellations with an empirical and a numerical approach. And since, by modelling constellation deformations, we ”simulate” expected deformations of a given object, we eliminate the need for a set of training images. This is consistent with the *one-shot learning* restriction, meaning, that for the model construction we require a single planar training image, depicting the object of interest. Referring back to our thought experiment of detecting the letter M, in geometric terms, it is clear that in the case of the letter M object, the bottom-left ”leg” feature lies opposite of the bottom-right ”leg” feature. This is strong prior knowledge which could potentially prevent mismatched features from negatively influencing the object detection process and achieve greater robustness against occlusion and background clutter.

The remainder of the thesis is structured as follows. An overview of related work is given in Chapter 2, followed by an overview of the computer vision approaches most relevant to our problem and proposed solutions in Chapter 3. A detailed description of the constellation model construction is given in Chapter 4, and a description of the object detection process utilizing the constructed model is given in Chapter 5. Chapter 6 gives an overview of the experiment methodology, implementation details and results, followed by conclusions in Chapter 7.

Chapter 2

Related work

Computer-vision-based object detection applies a range of approaches such as sliding window [12], template matching [13], image segmentation and blob analysis [14], bag-of-words models [15], and different feature-based object detection methods [11, 16, 2, 17]. In the following we provide a brief overview of these.

A standard technique for the detection of object categories is the sliding window method [12]. In the latter an image is scanned with a number of windows trough different scales and aspects, and each window is classified into pre-learned object categories. A drawback of this approach is the need for a training set of significant size in order to train the classification algorithms. Another drawback is the large number of windows, which are essentially various forms of filters across multiple scales, that need to be verified. This leads to a significant complexity of the approach, which makes it time-consuming at the least. Template matching [13] is a technique for locating parts of a test image which match a specific template image. This usually requires searching a large amount of locations in images, in order to determine the best-matching location. Template matching techniques rely solely on a template image of the object, which makes them inadequate in cases where objects in test images are deformed or occluded, i.e., diverge from the template image. Image segmentation and blob analysis [14] techniques require a segmentation step which segments an image into parts, whereby interesting parts of the image, i.e., blobs, undergo further processing. The segmentation step is crucial in this process, since the assumption is that the test image will be segmented in such a way, that segmented parts allow for a straight-forward extraction of interesting regions. Although good performance can be achieved in mostly homogeneous regions, the process is vulnerable to cluttered and textured areas, since these may get segmented into separate parts, blurring the distinction from interesting parts. Additionally, the performance of most segmentation algorithms depends highly on input parameters and varies with the scale of the object in the test image. Authors in [15] present the bag-of-keypoints method for efficient information extraction and classification. The feature-based approach uses local regions around interest points, defined by descriptors which are invariant to affine transformation, for object category classification. Utilizing these descriptors a classifier is trained using either the Naive Bayes [25] or the support vector machine [26] (SVM) method. However, the bag-of-keypoints method is an orderless representation, ignoring spatial relations between keypoints, and requires a training set of images in order to construct bags-of-keypoints which makes it inadequate for our intended purpose. Feature-based object detection methods utilize a variety of image features, extracted from images via feature detection algorithms [11, 16, 2, 17]. These algorithms rely on high-contrast regions, such as points and edges, in order to obtain interest points for which an corresponding descriptor is computed. The descriptor is used to match newly detected features to features representing the object. A method for robustly estimating multiple view relations from point correspondences, such as RANSAC [27] is applied to the set of matched features in order to find a subset of best matching features, which map the object of interest to a specific location in the test image. Since feature-based object detection methods preserve spatial relations between features in images, these are ideal for building constellations of features.

The scale-invariant feature transform [11] (SIFT) is a feature detection algorithm applied in state of the art feature-based object detection methods. The algorithm is designed to detect and describe local features in images, which are extracted in a cascade of scale-space extrema detection, keypoint localization, orientation assignment and the computation of keypoint descriptors. A SIFT feature consists of a detected keypoint, represented by its location, scale and orientation, and of a corresponding descriptor. Since the introduction of SIFT, a number of feature-detection algorithms based on SIFT with modifications have been published. These modifications include a rotation-invariant generalization of the SIFT descriptor, i.e., RIFT [19]; a robust general context descriptor encoding edge orientation, density and hue information, i.e., G-RIF [20]; a vector of image gradients computed in both 2D space directions within the support region with reduced dimensions as descriptor, i.e., PCA-SIFT [21]; an extension of the SIFT descriptor with gradient location-orientation histograms to enhance robustness and distinctiveness, i.e., GLOH [18]; a descriptor generated from a standard SIFT descriptor, by setting each histogram bin to its rank in a sorted array of bins, i.e., SIFT-Rank [22]. Authors in [18] conduct an extensive evaluation of different feature-detection algorithms and conclude that SIFT, and SIFT-like descriptors outperform other contemporary local descriptors, exhibiting the highest matching accuracies for affine transformations of regions. The evaluation suggests that SIFT, and SIFT-based feature descriptors are most robust and distinctive, and are thus best suited for feature description and matching. Not included in the evaluation of [18], authors in [16] propose a feature detection algorithm dubbed *Speeded-Up Robust Features*, i.e., SURF, which shares partial similarities with SIFT and is several times faster. SURF features, based on sums of $2D$ Haar wavelet responses and integral images, are less affected by different image transformations than SIFT, as claimed by the authors. A common aspect of all the above mentioned feature-detection algorithms is the feature representation. All features consist of a detected keypoint, represented by its location, scale, orientation, and of a corresponding descriptor.

Similar work on preserving spatial relations between local features can be found in [23, 24], where, for a given object, authors present an approach towards encoding triplets of neighbouring features and locating the encoded triplets across test images. The mass pipeline detection process shows promising results on the FlickrLogos-32 dataset [28]. However, the whole approach requires a set of diverse training images, whereby the problem of object deformations is addressed by warping either training images, or database images in order to achieve robustness against object deformations, i.e., deformations of spatial relations between features. We argue that the spatial layout encoding technique could be generalized to incorporate an arbitrary number of neighbouring features and spatial deformations of features, whereby requiring a single planar training image, i.e., one-shot learning.

One-shot learning of object categories in computer vision approaches can be found in [3], where authors present an approach towards learning visual models of object categories by probabilistic models. The posterior model of an object category is obtained by updating the prior, represented by a probability density function on model parameters, considering one or more observations. One-shot learning in the context of gesture and facial expressions can be found in [29], wheres authors in [30] deal with zero-shot learning in the context of visual object categories, i.e., the problem of object recognition with no training examples.

Chapter 3

Basic object detection theory

In this chapter, we describe computer vision methods which are used throughout the thesis, in order to provide a better understanding of the basic principles of feature-based object detection methods. This principles include transformation parameter estimation (Section 3.1), image feature extraction (Section 3.2), object localization (Section 3.3) and robust point-to-point correspondence estimation (Section 3.4).

3.1 Estimating transformation parameters

A commonly encountered task in computer vision is the problem of estimating transformation parameters between two sets of point correspondences. Given two sets of point correspondences, where x_q and y_p denote two corresponding points, the aim is to estimate the transformation parameters so that the mean square error between the two sets is minimized [31]. Considering a system of p linear equations and q unknowns, where a denotes scalar values

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + ... + a_{1q}x_q &= y_1 \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2q}x_q &= y_2 \\ &\vdots \\ a_{p1}x_1 + a_{p2}x_2 + ... + a_{pq}x_q &= y_p \end{aligned}$$

the system can be rewritten as $\boldsymbol{Ax} = \boldsymbol{y}$, where $\boldsymbol{A}$ denotes a $p \times q$ real matrix and $\boldsymbol{x} \in \mathbb{R}^q$ and $\boldsymbol{y} \in \mathbb{R}^p$ denote vectors of point correspondences

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1q} \\ a_{21} & a_{22} & \dots & a_{2q} \\ \vdots & & & \\ a_{p1} & a_{p2} & \dots & a_{pq} \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_q \end{bmatrix} \quad \text{and} \quad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_p \end{bmatrix}.$$

The mean square error to be minimized between point correspondences is given by

$$e(\boldsymbol{x}) = \sum_{i=1}^{p} (a_{i1}x_1 + \dots + a_{iq}x_q - y_i)^2 = ||\boldsymbol{Ax} - \boldsymbol{y}||_2^2,$$

where $|| \cdot ||_2$ denotes the Euclidean norm. When the matrix $\boldsymbol{A}$ is of full rank, i.e., $rank(\boldsymbol{A}_{p\times q}) = q$, the matrix $\boldsymbol{A}^T\boldsymbol{A}$ is invertible and the equation rearranges to

$$\begin{aligned} \boldsymbol{Ax} &= \boldsymbol{y}, \\ \boldsymbol{x} &= (\boldsymbol{A}^T\boldsymbol{Ax})^{-1}\boldsymbol{Ax}^T\boldsymbol{y}, \\ \boldsymbol{x} &= \boldsymbol{A}^\dagger\boldsymbol{y}, \end{aligned}$$

where

$$\boldsymbol{A}^\dagger = (\boldsymbol{A}^T\boldsymbol{Ax})^{-1}\boldsymbol{Ax}^T$$

denotes the Moore–Penrose pseudoinverse. Alternatively, the solution can be obtained by singular value decomposition of the matrix $\boldsymbol{A}$, i.e.,

$$\boldsymbol{A} \overset{\text{svd}}{=} \boldsymbol{USV}^T.$$

The matrix of right-singular vectors $\boldsymbol{V}^T$ is a $q \times q$ orthogonal matrix, and it's columns represent the eigenvectors of $\boldsymbol{A}^T\boldsymbol{A}$. The last eigenvector of $\boldsymbol{V}^T$, corresponding to the smallest eigenvalue, holds the estimated parameters.

It is important to note, that, depending on the number of available correspondences, different transformation parameters can be estimated. Table 3.1 gives a brief summary of the number of correspondences required in order to estimate the transformation parameters of a certain transformation, and the required rank of the matrix $\boldsymbol{A}$, to ensure that the estimated parameters are not degenerated.

Table 3.1: Estimating transformation parameters.

Transformation	Number of correspondences	Matrix rank
Projective	4	8
Affine	3	6
Similarity	2	4

3.2 Scale Invariant Feature Transform

The scale-invariant feature transform [11] (SIFT) is a feature detection algorithm applied in state of the art feature-based object detection methods. The algorithm is designed to detect and describe local features in images. SIFT features are extracted from images in a cascade of scale-space extrema detection, keypoint localization, orientation assignment and the computation of keypoint descriptors. A SIFT feature thus consists of a detected keypoint, represented by its location, scale and orientation, and a corresponding descriptor.

3.2.1 Scale-space extrema detection

The scale space of a given image $I(x, y)$, is defined as a function $L(x, y, \sigma)$, produced from the convolution of a variable-scale Gaussian $G(x, y, \sigma)$ with the given image, where

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

where $*$ is the convolution operation and

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{(x^2+y^2)/2\sigma^2}.$$

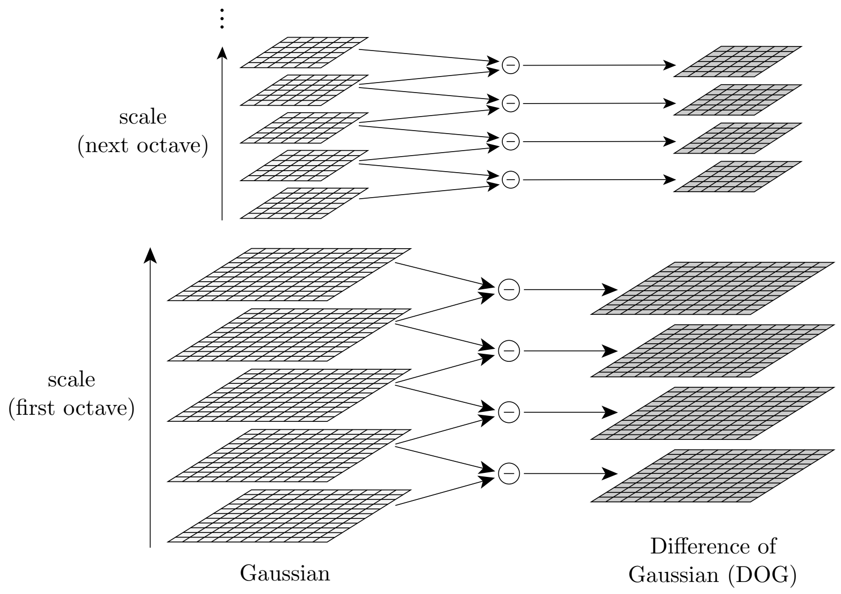


Figure 3.1: Image obtained from [11]. The initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. The difference-of-Gaussian images, on the right, are produced by subtracting adjacent Gaussian images. After each octave, the Gaussian image is down-sampled by a factor of two. This process is repeated for each octave of the scale space.

Stable keypoint locations in scale space are efficiently detected using scale-space extrema in the difference-of-Gaussian (DoG) function, illustrated in Figure 3.1, convolved with the image $D(x, y, \sigma)$, which is computed from the difference of two nearby scales separated by a constant multiplicative factor k, i.e.,

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned}$$

3.2.2 Keypoint localization

The local maxima and minima of $D(x, y, \sigma)$ are detected by comparing each sample point to its eight neighbours in the current image and nine neighbours

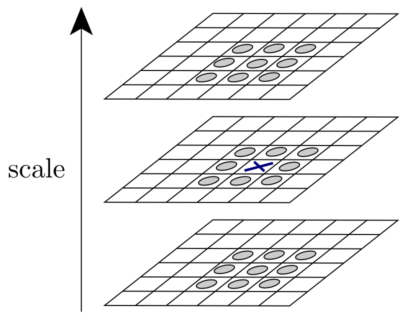


Figure 3.2: Image obtained from [11]. The extrema of the difference-of-Gaussian images, i.e., the maxima and minima, are detected by comparing a pixel, marked with $\boldsymbol{\chi}$, to its 26 neighbours, depicted by gray circles, in a 3×3 region at the current and adjacent scales.

in the scale above and below, as illustrated in Figure 3.2. The point is selected as an extrema only if it is larger or smaller than all of its neighbours. Low contrast or poorly localized candidates along edges present outliers and are rejected, by computing the function value of D at the extremum $\hat{\boldsymbol{\chi}}$. By Taylor series expansion, the scale-space function D rearranges to:

$$D(\boldsymbol{\chi}) = D + \frac{\partial D}{\partial \boldsymbol{\chi}}^T \boldsymbol{\chi} + \frac{1}{2}\boldsymbol{\chi}^T \frac{\partial^2 D}{\partial \boldsymbol{\chi}^2}\boldsymbol{\chi}, \quad \text{where} \quad \boldsymbol{\chi} = (x, y, \sigma)^T.$$

The location of the extrema $\hat{\boldsymbol{\chi}}$ is determined by taking the derivative of the function D with respect to $\boldsymbol{\chi}$ and setting it equal to zero, yielding:

$$\hat{\boldsymbol{\chi}} = -\frac{\partial^2 D}{\partial \boldsymbol{\chi}^2}^{-1} \frac{\partial D}{\partial \boldsymbol{\chi}}.$$

By substituting $\hat{\boldsymbol{\chi}}$ into D the equation rearranges to:

$$D(\hat{\boldsymbol{\chi}}) = D + \frac{1}{2}\frac{\partial D}{\partial \boldsymbol{\chi}}^T \hat{\boldsymbol{\chi}}.$$

The value of $D(\hat{\boldsymbol{\chi}})$ at extrema regions must be greater than a threshold τ_{extrema}, i.e., $|D(\hat{\boldsymbol{\chi}})| > \tau_{\text{extrema}}$, else the extrema point $\hat{\boldsymbol{\chi}}$ is considered to be unstable and is rejected.

The DoG function has strong responses along edges, even if the location along the edge is poorly determined, yielding unstable extrema points. These are further filtered out by computing principle curvatures at each extrema point, since unstable points exhibit a large principal curvature across the edge and a small one in the perpendicular direction. Principal curvatures are computed at each extrema point from a 2×2 Hessian matrix $\boldsymbol{H}$, i.e.,

$$\boldsymbol{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}.$$

Outliers are removed by evaluating the trace and the determinant of the matrix $\boldsymbol{H}$, which correspond to the summation and the product of the eigenvalues of $\boldsymbol{H}$, which in turn are proportional to the principal curvatures of D. The trace and determinant of $\boldsymbol{H}$ are computed as:

$$\begin{aligned} Tr(\boldsymbol{H}) &= D_{xx} + D_{yy} = \alpha + \beta, \\ Det(\boldsymbol{H}) &= D_{xx}D_{yy} - D_{xy}^2 = \alpha\beta. \end{aligned}$$

If the computed determinant is negative, the curvatures have different signs and the extrema point is rejected. Let r denote the ratio between the largest and the smallest magnitude eigenvalue. Then it follows that $\alpha = r\beta$, and

$$\frac{Tr(\boldsymbol{H})^2}{Det(\boldsymbol{H})} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r}.$$

The obtained quantity depends only on the ratio of the eigenvalues, reaching a minimum when the two eigenvalues are equal, and increasing with r. In order to check that the ratio of principle curvatures is below a threshold r, and determine the stability of the extrema, the following term is evaluated

$$\frac{Tr(\boldsymbol{H})^2}{Det(\boldsymbol{H})} < \frac{(r+1)^2}{r}.$$

3.2.3 Orientation assignment

Each localized keypoint is assigned a consistent orientation, based on local image properties, achieving invariance to image rotation. To ensure that computations are scale-invariant, the scale of the keypoint is used to select the Gaussian-smoothed image L with the closest scale. At the selected scale, the gradient magnitude $m(x, y)$, and orientation $\theta(x, y)$, are precomputed for each image sample $L(x, y)$ using pixel differences

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1)^2},$$
$$\theta(x,y) = \tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right).$$

Within a region around the keypoint, an orientation histogram with 36 bins is formed, covering the 360 degree range of gradient orientations. Each sample in the histogram is weighted by its gradient magnitude and a Gaussian-weighted circular window with a $\sigma = 1.5s$, where s is the scale of the keypoint. Histogram peaks represent dominant gradient directions. The highest peak, along with peaks within 80% of the highest peak, are used to create keypoints with the given orientations. Therefore, for the same location and scale, multiple keypoints with different orientations are created, in the case of multiple peaks with similar magnitude.

3.2.4 Descriptor computation

All previous operations impose a repeatable 2D coordinate system to a local image region, by assigning an image location, scale and orientation to each keypoint, providing invariance to these parameters. For each keypoint, a distinctive descriptor that is invariant to variations such as illumination change or 3D viewpoint change is computed.

The image gradient magnitude and orientations are sampled around a keypoint location in a 16×16 neighborhood, using the scale of the keypoint to select the level of Gaussian blur for the image. Orientation invariance is achieved by rotating the coordinates of the descriptor and the gradient orientations relative to the keypoint orientation. A Gaussian weighting function with σ equal to one half of the width of the descriptor window is used to assign a weight to the magnitude of each sample point. The Gaussian function serves to omit small position changes of the window, and reduce influence of gradients further away from the center of the descriptor. The

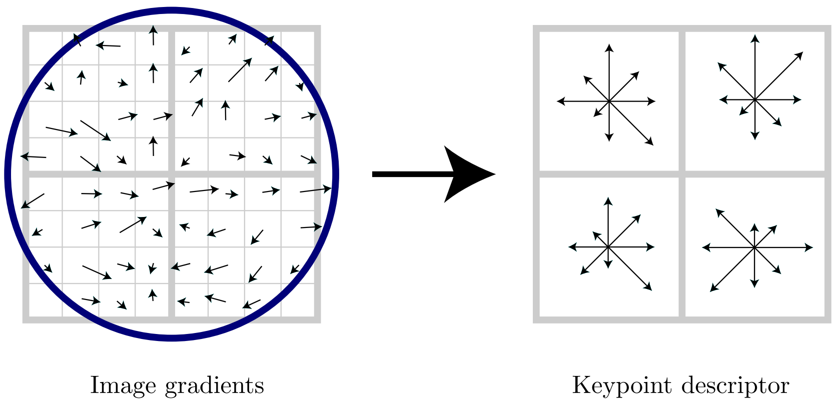


Figure 3.3: Image obtained from [11]. A keypoint descriptor is created by computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, weighted by a Gaussian window, depicted by a blue circle as shown on the left. These samples are in turn accumulated into orientation histograms, which summarize the contents over 4×4 subregions, as shown on the right. The length of each arrow corresponds to the sum of the gradient magnitudes near that direction within the region. The figure illustrates a 2×2 descriptor array computed from an 8×8 set of samples, whereas SIFT uses 4×4 descriptors computed from a 16×16 set of samples.

descriptor is obtained from a 4×4 array of histograms with 8 orientation bins in each histogram, containing the values of the magnitudes of the peaks. By concatenating these 16 histograms, with 8 orientations each, a 128 dimensional vector representation of the descriptor is formed. The computation of the keypoint descriptor is illustrated in Figure 3.3. The final steps include normalizing the vector to unit length, in order to account for illumination changes. Unit vector values are further bound to a maximum value of 0.2, and the bound unit vector is re-normalized, emphasizing the distribution of orientations rather than orientation peaks. This is necessary in order to address the influence of large gradient magnitudes, which when normalized, strongly reduce the influence of the smaller gradient magnitudes.

3.2.5 Keypoint matching

SIFT features are matched to precomputed SIFT features via the corresponding descriptors in terms of the nearest-neighbour principle. This means that each detected keypoint $\boldsymbol{k}_i$ is associated to the closest precomputed keypoint $\boldsymbol{m}_i$ in terms of the smallest Euclidean norm between the descriptors $\boldsymbol{d}_i^k$ and $\boldsymbol{d}_i^m$, i.e.,

$$\boldsymbol{k}_i \to \boldsymbol{m}_i \quad \text{if} \quad \boldsymbol{d}_i^m = \arg\min_{\boldsymbol{d}_i^m}(||\boldsymbol{d}_i^k - \boldsymbol{d}_i^m||_2).$$

The Euclidean norm between two given n-dimensional vectors $\boldsymbol{d}^k$ and $\boldsymbol{d}^m$ is defined as:

$$e(\boldsymbol{d}^k, \boldsymbol{d}^m) = ||\boldsymbol{d}^k - \boldsymbol{d}^m||_2 = \sqrt{\sum_{i=1}^{n}(\boldsymbol{d}_i^k - \boldsymbol{d}_i^m)^2}.$$

However, many extracted features will not have any correct match with the precomputed features, due to background clutter and noise. This is accounted for by looking at the ratio of the distances between the closest and the second closest match. A potential feature match $\boldsymbol{k}_i \to \boldsymbol{m}_i$ is rejected, if the similarity threshold between the detected feature $\boldsymbol{k}_i$ and two precomputed features $\boldsymbol{m}_i$ and $\boldsymbol{m}_j$ is higher than a threshold θ, i.e., $\frac{s_1}{s_2} > \theta$, where $s_1 = \boldsymbol{k}_i \sim \boldsymbol{m}_i$ and $s_2 = \boldsymbol{k}_i \sim \boldsymbol{m}_j$.

As argued in the introduction (Chapter 1), this threshold represents a potential problem, since it does not only reject non-discriminative features, but also rejects potential useful matches, causing a loss of valuable information.

3.3 The Generalised Hough Transform

In object detection tasks, a commonly used method for localizing objects in images is the Generalised Hough Transform [32] (GHT). The Generalised Hough Transform is a generalization of the Hough Transform [33] method, which was initially designed for the detection of objects described with an parametric function, such as lines, circles or ellipses. The Generalised Hough

Transform is modified to use the principle of template matching, enabling it for the detection of an arbitrary object described with it's model, instead of only objects described with an analytical function.

In order to utilize the Generalised Hough Transform method for the detection of an arbitrary object, i.e., shape in a test image, a model of the shape, i.e., object needs to be constructed beforehand. Given an arbitrary shape with a fixed reference point, as shown in Figure 3.4, the information provided by the boundary points is used to construct the R-table, which acts as a transformation mechanism. For each boundary point the gradient direction θ and the vector $\boldsymbol{r}$ pointing to the reference point X is computed. In the R-table (Table 3.2), the vector $\boldsymbol{r}$ is stored as a function of θ. Having computed this for each boundary point, the R-table acts as a model, representing the shape, i.e., object. The GHT localizes the object model in the

Table 3.2: The R-table.

θ	$R(\theta)$
θ_a	$\boldsymbol{r}_a$
θ_b	$\boldsymbol{r}_b$
θ_c	$\boldsymbol{r}_c$

test image, by accumulating votes for potential model locations in a matrix $\boldsymbol{A}$ of size $p \times q$, called the *accumulator array.* In general, the ratio between p and q equals the ratio of the height and width of the test image, but the accumulator array may be of smaller resolution.

For a given boundary point $E(x, y)$, detected on a test image, the gradient direction θ_e is computed and a lookup is performed in the R-table. Let $R(\theta_e) = \boldsymbol{r}$, the boundary point thus casts a vote in the accumulator array by incrementing the accumulator cell $\boldsymbol{A}(E_x+\boldsymbol{r}_x, E_y+\boldsymbol{r}_y)$ that corresponds to the point $E+\boldsymbol{r}$. If the accumulator array is smaller than the original test image, the point $E + \boldsymbol{r}$ is scaled down accordingly. In order to address arbitrary object orientations, α, and scales, s, two more parameters are added to the

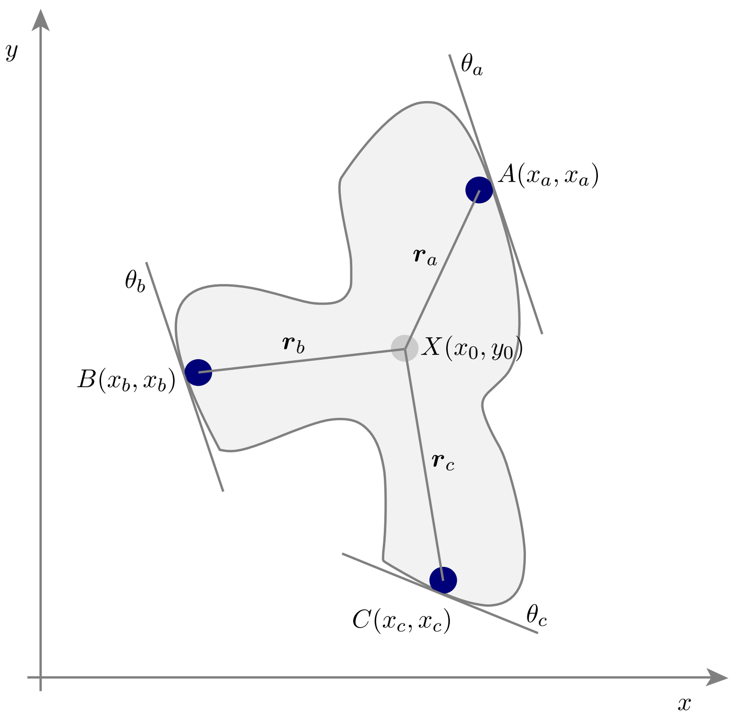


Figure 3.4: An illustration of an arbitrary shape with a fixed reference point X and three boundary point A, B and C.

model description and two more dimensions to the accumulator array. Thus, for each boundary point the gradient direction θ is computed, and for each resulting evaluation $\boldsymbol{r}$ of the function $R(\theta)$ and all values of α and s, x' and y' are computed as:

$$\begin{aligned} x' &= s(x \cos\alpha - y \sin\alpha), \\ y' &= s(x \sin\alpha - y \cos\alpha), \end{aligned}$$

and the corresponding accumulator cell $\boldsymbol{A}(x', y', s, \alpha)$ is incremented. The object location in the test image is indicated by peaks in the accumulator array, since the assumption is that votes from boundary points located on the object will be accumulated in the same cell, distinguishing it from other cells.

3.3.1 SIFT-based model

The Generalised Hough Transform converts the problem of localizing an object in a test image, to the problem of determining the transformation parameters which map the object model to a specific location in the test image. Determining the transformation parameters thus indirectly determines the location of the object in the test image.

Let $\mathcal{M}$ denote a set of SIFT features, detected on an arbitrary object of interest. These features, and for each feature a vector pointing to a reference point on the object, form an abstract representation of the object, i.e., encode positions of features relative to the object center, forming a model of the object. Let $\mathcal{K}$ denote the set of SIFT features, detected on a test image containing the object of interest. When each detected feature $\boldsymbol{k}_i \in \mathcal{K}$ is matched to a corresponding model feature $\boldsymbol{m}_i \in \mathcal{M}$, a vote is cast in the accumulator array by computing the transformation $\boldsymbol{T_A} : \boldsymbol{m}_i \rightarrow \boldsymbol{k}_i$, which maps a given model feature to a given detected feature. The computed transformation is used to map the object reference point to the accumulator array in which the corresponding cell is incremented. In general, the transformation $\boldsymbol{T_A}$ can be estimated by computing a least-squares fit solution which determines the transformation parameters from a given model feature $\boldsymbol{m}_i$, to a given detected feature $\boldsymbol{k}_i$, i.e., $\boldsymbol{m}_i \cdot \boldsymbol{T_A} = \boldsymbol{k}_i$.

Due to noise interference during the voting process, votes of features belonging to the object are not likely to be accumulated in the same cell, but may get scattered around the target cell, accumulating votes in the neighbouring cells. This, in turn, introduces noise in the localization process and reduces the capability of distinguishing cells that indicate object presence from those that do not. In order to address this drawback, a common practice is to apply non-maxima suppression to the accumulator array, suppressing all cells that are not part of a local maxima, i.e., suppressing a given cell if an immediate neighbouring cell accumulated more votes. Cells with the highest number of accumulated votes in the processed accumulator array indicate a potential object location in the test image.

3.4 The Sample Consensus

Given a set of detected features which casted a vote into the same cell in the accumulator array, the object location in the test image can be determined by finding the transformation parameters which map the set of model features to the set of detected features. Since each detected feature is associated with a model feature, the transformation parameters can be estimated by considering two sets of point correspondences, i.e., $\boldsymbol{T} : A \rightarrow B$, where $A \subset \mathcal{M}$ and $B \subset \mathcal{K}$.

However, in practice there is no guarantee that the two sets are correctly matched, meaning that matches contain outliers. Simply determining the transformation parameters with a least squares method thus results in noisy parameter estimation, since general least squares methods do not account for outlier influence. In order to identifying inliers and assure a robust transformation parameter estimation, even in cases with a significant number of outliers, the Maximum Likelihood Estimation Sample Consensus [34] (MLESAC) is applied. The MLESAC is essentially an improved version of the Random Sample Consensus [27] (RANSAC) algorithm.

RANSAC is an iterative, stochastic algorithm that consists of two steps: (i) hypothesis generation and, (ii) hypothesis evaluation. The algorithm generates a hypothesis by randomly selecting a minimal subset of point correspondences $B' \subset B \models A' \subset A$ in order to estimate a solution $\boldsymbol{T} : A' \rightarrow B'$, and proceeds with the evaluation of the generated solution for support from the complete set. The support is measured in terms of the number of inliers, i.e., the number of points with error e below a given threshold τ_{inliers}. The solution $\boldsymbol{T}$ is re-estimated by least-squares, taking all inliers into account. The RANSAC iteratively repeats these steps and returns the estimated solution with the highest support, i.e., the maximum number of inliers. The error of a given point $\boldsymbol{b}_i \in B$ and it's transformed correspondence point $\hat{\boldsymbol{a}}_i \in \hat{A}$ is measured as the Euclidean norm between the two points, i.e.,

$$e(\boldsymbol{b}, \hat{\boldsymbol{a}}) = ||\boldsymbol{b} - \hat{\boldsymbol{a}}||_2 = \sqrt{(\boldsymbol{b}_x - \hat{\boldsymbol{a}}_x)^2 + (\boldsymbol{b}_y - \hat{\boldsymbol{a}}_y)^2}.$$

Let n denote the minimum number of data points required to estimate a solution, and let w the fraction of correctly matched correspondences. The expected number of required iterations in order to obtain all inliers with a high probability α equals to

$$N = \frac{log(1-\alpha)}{log(1-w^n)}.$$

In contrast to RANSAC, whose aim is to maximize the number of inliers that support an estimated solution, MLESAC aims to maximize the likelihood of the solution. The hypothesis likelihood is estimated by representing the error probability distribution for the entire set as a mixture of a Gaussian distributions for inliers and a uniform distribution for outliers, i.e.,

$$Pr(e) = \gamma \frac{1}{\sqrt{2\pi\sigma^2}} \exp\Big(-\frac{e^2}{2\sigma^2}\Big) + (1-\gamma)\frac{1}{v},$$

where the first term models the likelihood that a point is an inlier by

$$p_{inlier} = \gamma \frac{1}{\sqrt{2\pi\sigma^2}} \exp\Big(-\frac{e^2}{2\sigma^2}\Big),$$

and the second term the likelihood that a point is an outlier by

$$p_{outlier} = (1-\gamma)\frac{1}{v}.$$

Here, γ denotes the mixing parameter, v a constant, and σ the standard deviation of the error on each coordinate. The determination of the parameters γ and v requires some approximate knowledge of the outlier distribution in the set. The error minimized by MLESAC is the negative log-likelihood, where n denotes the number of correspondences,

$$\mathcal{L} = -\sum_i \log\Bigg(\gamma\Big(\frac{1}{\sqrt{2\pi\sigma^2}}\Big)^n \exp\Big(-\frac{e^2}{2\sigma^2}\Big) + (1-\gamma)\frac{1}{v}\Bigg).$$

Chapter 4

The constellation model

From an image depicting an arbitrary object, a feature-based model representation $\mathcal{M}$ of the object is constructed, utilizing detected image features. For each feature, the spatial relations between the neighbouring features are encoded, obtaining a constellation. For each constellation, expected variations of features are modelled in order to obtain deformable constellations, which are able to address deformations of local regions. All deformable constellations are encoded in a common space, dubbed *the unit space*. Similar variations of features in the unit space are merged in order to reduce the total number of encoded variations. Additionally, the object center point, the object bounding-box, and an intensity template of the object are stored in the model.

In this chapter, the process of encoding keypoints in the Cartesian coordinate system is given in Section 4.1. Section 4.2 gives a detailed description of encoding keypoint constellations, while Section 4.3 describes how constellation deformations are modelled. Section 4.4 describes the process of consolidating encoded keypoint constellations, while the constructed constellation model of an object is presented in Section 4.5.

4.1 Encoding keypoints

In the polar coordinate system a keypoint is represented by a position vector $[x, y]^T$, which encodes the absolute location of the keypoint in an image, the scale s and the angular coordinate θ, i.e., $\boldsymbol{k}^{\mathcal{P}} = [x, y, s, \theta]^T$. Thus, in the Cartesian coordinate system, a keypoint can be encoded by a pair of points $\boldsymbol{k}^{\mathcal{C}} = [\boldsymbol{c}, \boldsymbol{o}]^T$, whereby, $\boldsymbol{c} = [x, y]^T$, representing the position of the keypoint and $\boldsymbol{o} = [s \cdot \cos(\theta), s \cdot \sin(\theta)]^T$, representing the scale and angle. For further processing, all keypoints are assumed to lie in Cartesian space, so the upper $\mathcal{P}$ and $\mathcal{C}$ notations are left out from here on.

4.2 Encoding constellations of keypoints

A keypoint constellation consists of a set of keypoints, modelling local object structures. For each keypoint, the corresponding descriptor and the spatial relations between the neighbouring keypoints are encoded. From here on, we distinguish between two terms denoting keypoints, i.e., *pair keypoints*, denoted with $\boldsymbol{p}$ and *root keypoints*, denoted with $\boldsymbol{r}$. A pair keypoint $\boldsymbol{p}$ is considered to be in the neighbourhood of a given root keypoint $\boldsymbol{r}$, if the distance between $\boldsymbol{r}$ and $\boldsymbol{p}$ is less than ϵ_{max}. In order to prevent cases where $\boldsymbol{p}$ is arbitrary close to $\boldsymbol{r}$, an additional constraint is that the distance between $\boldsymbol{r}$ and $\boldsymbol{p}$ must be greater than ϵ_{min}. The region between ϵ_{min} and ϵ_{max} is dubbed *the ϵ region.* It follows that a keypoint is considered to be a pair keypoint $\boldsymbol{p}$ of a given root keypoint $\boldsymbol{r}$, if $\boldsymbol{p}$ lies within the ϵ region around $\boldsymbol{r}$, as illustrated in Figure 4.1.

The ϵ region is determined by considering the sacle s of the root keypoint. Any detected keypoint, which lies within the two thresholds $\epsilon_{min} = s\alpha$ and $\epsilon_{max} = s\beta$ is considered to be in the neighborhood of $\boldsymbol{r}$, i.e., a pair keypoint of $\boldsymbol{r}$. The threshold ϵ_{min} serves to prevent cases in which keypoints located too close to the root keypoint are encoded as neighbours. Accordingly, ϵ_{max} serves to prevent cases, where keypoints located too far from the root keypoint are encoded as neighbours. The two parameters α and β are determined as follows. The value of α is set to 0.5, meaning that only keypoints which are located at least half of the root keypoint scale away from the center are considered as pairs. The value of β is determined by taking into account the maximal distance d_{max}, between two detected keypoints, since it roughly models the object size. Since we would like to ensure a good ϵ region estimation for objects of arbitrary sizes, the value of β is set to be a fraction of the estimated distance, i.e., $\beta = \tau_{\text{dst}} \cdot d_{max}$.

Since $\boldsymbol{r}$ represents the root of the constellation, all of the corresponding pairs pairs are encoded relative to $\boldsymbol{r}$. The encoding is performed by computing a similarity transformation $\boldsymbol{T}_{\mathcal{U}} : \boldsymbol{r} \rightarrow \boldsymbol{u}$, which maps $\boldsymbol{r}$ from the image space $\mathcal{I}$, to the so called unit keypoint $\boldsymbol{u}$, which lies in the unit space, i.e., $\boldsymbol{u} \in \mathcal{U}$. In the unit space, the unit keypoint lies in the cartesian coordinate system origin with a scale of one and a zero angle, i.e., $\boldsymbol{u} = [0, 0, 1, 0]^T$. Each pair keypoint $\boldsymbol{p}$ is mapped to the unit space of the root keypoint by the transformation $\boldsymbol{T}_{\mathcal{U}}$. All pair keypoints are thus encoded relative to the root keypoint.

4.3 Modelling constellation deformations

When an object is deformed, local structures retain their spatial relations depending on the degree of deformation. A keypoint constellation consisting of pair keypoints, encoded in the root keypoint unit space, rigidly models the spatial relations between the encoded keypoints. And since keypoints are merely points, i.e., $\boldsymbol{k} = [\boldsymbol{c}, \boldsymbol{o}]^T$, it follows that the encoded constellation does not address object deformations. In real-world scenarios, however, objects will likely be subject to different deformations. An encoded constellation must thus be able to address these expected deformations.

When an object deforms, keypoint locations will change. It follows that a deformable constellation can be obtained by considering the variance of the keypoint position. In order to model the expected variance of a keypoint within a constellation, we considered two different approaches. The

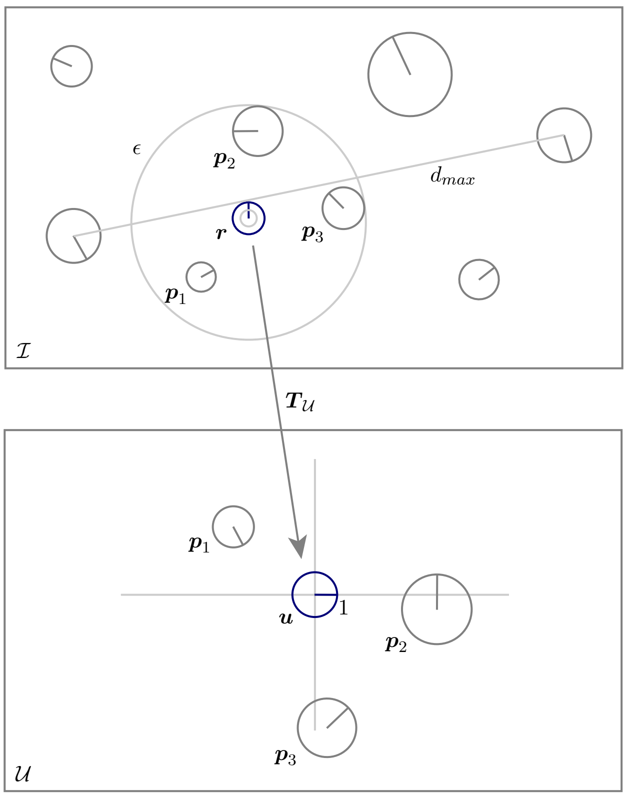


Figure 4.1: Keypoints in the image space $\mathcal{I}$, which lie within an ϵ region around a root keypoint $\boldsymbol{r}$ are considered to be the pair keypoints. The transformation $\boldsymbol{T}_{\mathcal{U}} : \boldsymbol{r} \rightarrow \boldsymbol{u}$ maps all pair keypoints to the normalized unit space $\mathcal{U}$, encoding a local keypoint neighborhood of the root keypoint.

first approach is to measure the expected variance of a keypoint by taking a set of images and subjecting them to a series of projective transformations i.e., warping the images. The expected variance of a keypoint position is measured by considering each detected keypoint on each warped image. The second approach is to model the expected variance by considering the maximal allowed deformation of an image, under which a keypoint detection method is capable of detecting a keypoint. The expected keypoint variance can thus be modelled by a function, obtained by considering the maximal allowed image deformation. The former and latter approaches, essentially modelling expected keypoint variance with a Gaussian, are presented in Sections 4.3.1 and 4.3.2, respectively.

4.3.1 Empirical keypoint variance

Given a set of planar images, depicting different objects, keypoint variance can be measured by warping these images and considering the variance of each keypoint detected on a warped image, relative to a reference keypoint, detected on a planar image. A planar image I^P is warped by a transformation $\boldsymbol{H}$, which transforms the planar image to a warped image, i.e., $\boldsymbol{H} : I^P \to I^H$, as illustrated in Figure 4.2. Given a reference keypoint $\boldsymbol{k}^P$, detected on a planar image, it's variance is measured by considering it's warped instance $\boldsymbol{k}^H$.

A warped keypoint $\boldsymbol{k}^H$, detected on a warped image is mapped back to the planar image by the transformation $\boldsymbol{H}^{-1}$, where the backprojected keypoint location and orientation will slightly vary compared to the reference keypoint. The backprojected warped keypoint $\hat{\boldsymbol{k}}^H$ is mapped to the unit space by the transformation $\boldsymbol{T} : \boldsymbol{k}^P \to \boldsymbol{u}$, where the center and angular points are recorded. If this is repeated for a set of keypoints, and a set of transformations, the measured data yields the expected variance of an arbitrary keypoint center and angular point, as shown in Figure 4.3.

For both, the measured center and angular data points, the variance is modelled in two steps: (i) retain 90% of the measured data, (ii) model

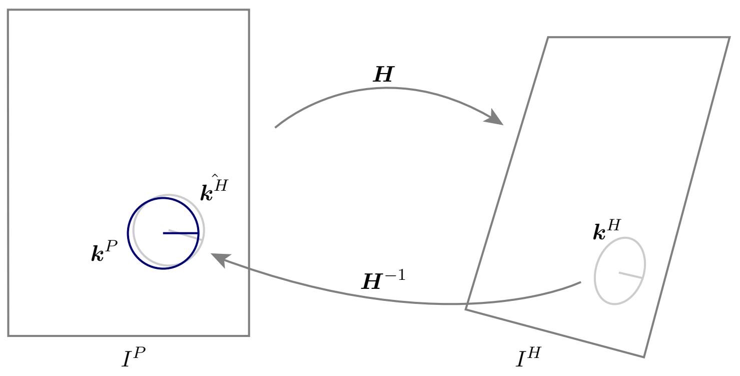


Figure 4.2: Give a transformation $\boldsymbol{H}$, the planar image I^P can be warped, i.e., subjected to projective transformation. Since $\boldsymbol{H}$ is known, a keypoint $\boldsymbol{k}^H$, depicted in gray, detected on a warped image can be backprojected by $\boldsymbol{H}^{-1}$, and it's variation determined relative to it's reference keypoint $\boldsymbol{k}^P$, depicted in blue.

the remaining data. At first, a Gaussian is fitted to the measured data by computing a covariance matrix Σ. Given a matrix of $2D$ data points $\boldsymbol{A}_{n\times 2} = [\boldsymbol{x}, \boldsymbol{y}]$, where $\boldsymbol{x} = [x_1 \dots x_n]^T$ denotes the x coordinates and $\boldsymbol{y} = [y_1 \dots y_n]^T$ denotes the y coordinates of the data points, the Gaussian of the data is modelled by the multivariate normal distribution $\mathcal{N}_{\mathcal{A}}(\mu, \Sigma)$, where μ represents the mean of the data and Σ the covariance matrix,

$$\begin{aligned} \mu &= (\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}), \\ \Sigma &= \begin{bmatrix} \mathrm{Var}(\boldsymbol{x}) & \mathrm{Cov}(\boldsymbol{x}, \boldsymbol{y}) \\ \mathrm{Cov}(\boldsymbol{x}, \boldsymbol{y}) & \mathrm{Var}(\boldsymbol{y}) \end{bmatrix} = \frac{1}{n}(\boldsymbol{A}^T \boldsymbol{A}), \end{aligned}$$

where

$$\begin{aligned} \bar{\boldsymbol{x}} &= \frac{1}{n}\sum_{i=1}^{n} x_i, \\ \mathrm{Var}(\boldsymbol{x}) &= \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{\boldsymbol{x}})^2, \\ \mathrm{Cov}(\boldsymbol{x}, \boldsymbol{y}) &= \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{\boldsymbol{x}})(y_i - \bar{\boldsymbol{y}}). \end{aligned}$$

All data points, exceeding a standard deviation of $\sigma \approx 1.65$ around μ are discarded in order to omit noise influence on variance estimation, leaving approximately 90% of characteristic data points for further processing. A detailed description of the verification, weather a give point lies within a given multivariate normal distribution is provided in Section 4.4. On the remaining data points, a Gaussian is re-fitted obtaining the final variance model, as illustrated Figure 4.4.

4.3.2 Analytical keypoint variance

Assume that a feature detection method is capable of detecting keypoints under affine transformations of γ degrees of the image. Keypoint localization will slightly vary, depending on γ, and will result in a variance of the keypoint location. The expected location variance can be modelled as follows. Imagine a plane Π with a unit keypoint $\boldsymbol{u}$, and a point $\boldsymbol{q}$ on the x-axis. By transforming Π with a transformation $\boldsymbol{T}_\gamma : \Pi \rightarrow \Pi'$, we simulate the maximal affine transformation by γ degrees, under which a keypoint can be detected, as shown in Figure 4.5.

The transformation $\boldsymbol{T}_u : \boldsymbol{u}' \rightarrow \boldsymbol{u}$ maps the transformed unit keypoint $\boldsymbol{u}'$ back to its initial position. By using $\boldsymbol{T}_u$ to back-project the transformed point $\boldsymbol{q}'$, the back-projection error between $\boldsymbol{q}$ and $\boldsymbol{q}''$ can be computed, i.e., $e = ||\boldsymbol{q} - \boldsymbol{q}''||_2$. Since the error depends on the distance of the point $\boldsymbol{q}$ from the coordinate system origin, the back-projection error has to be computed for n different points $\boldsymbol{q}$, at different distances. The measured back-projection error data yields a second degree polynomial function shown in Figure 4.6. From the measured data, we can extract the analytical relation between γ and the variance of a point by solving the following polynomial fit equation,

$$\boldsymbol{e} = \boldsymbol{D}\boldsymbol{c},$$

where the vector $\boldsymbol{e} \in \mathbb{R}^n$ corresponds to the backprojected errors of each point, the matrix $\boldsymbol{D}_{n\times 3}$ to the distances of each point from the coordinate system origin and the vector $\boldsymbol{c} \in \mathbb{R}^n$ to the second degree polynomial co-

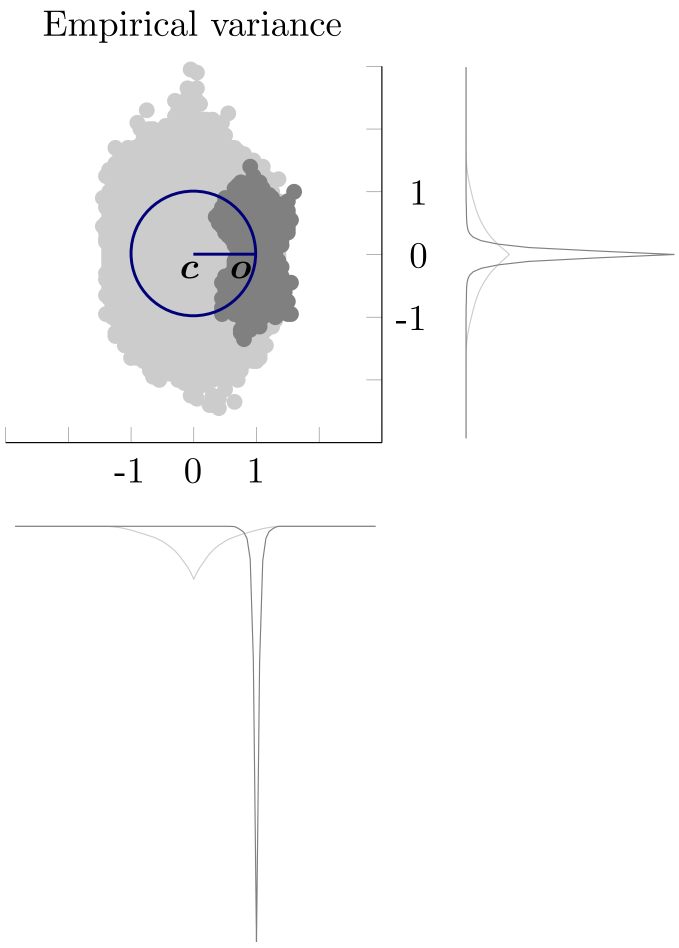


Figure 4.3: Expected keypoint variance, measured from a set of six images, each subjected to 1425 projective transformations in x, y and z (rotation) dimensions. The light gray color depicts the variance of the center point $\boldsymbol{c}$, while the dark gray color the variance of the angular point $\boldsymbol{o}$.

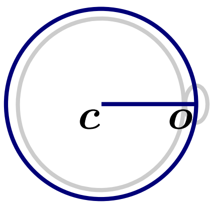


Figure 4.4: Modelled keypoint location and orientation variance by a Gaussian with $\sigma = 1$, depicted by a gray circle and ellipse, respectively.

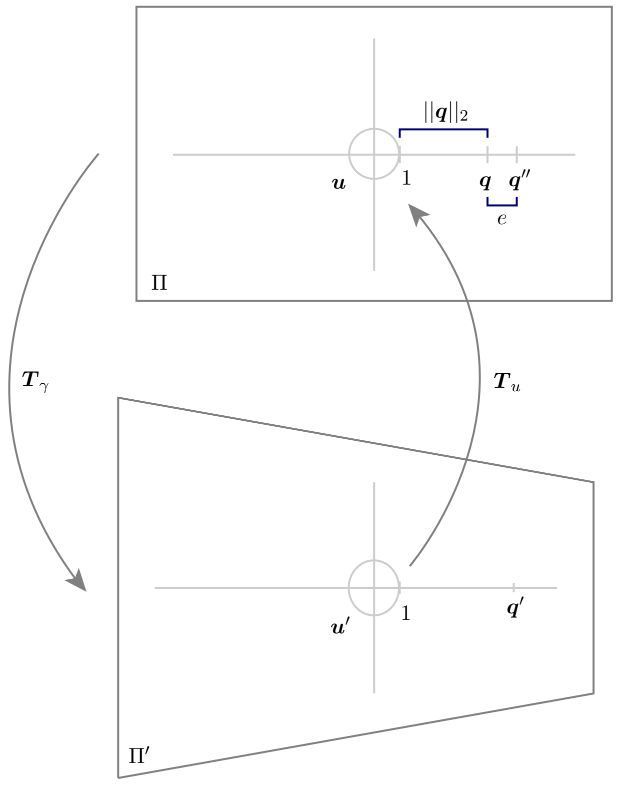


Figure 4.5: The plane Π is subjected to the maximal allowed affine transformation of γ degrees, under which a keypoint detection method is capable of detecting a keypoint. By reprojecting the transformed point $\boldsymbol{q}'$ back to the original plane Π, the reprojection error, relative to $\boldsymbol{q}$, can be measured.

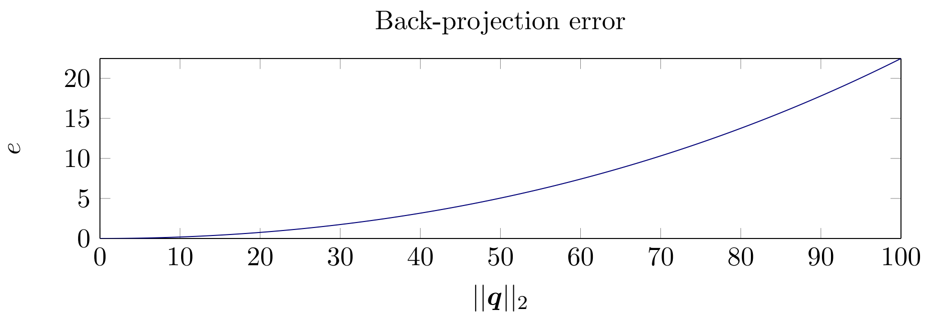


Figure 4.6: A second degree polynomial function, obtained from measuring the reprojection error of the point $\boldsymbol{q}$. The reprojection error e increases with the increasing distance of the point $\boldsymbol{q}$ from the coordinate system origin.

efficients. Using the Moore-Penrose pseudoinverse the equation rearranges to,

$$\boldsymbol{c} = (\boldsymbol{D}^T\boldsymbol{D})^{-1}\boldsymbol{D}^T\boldsymbol{e},$$

and the solution yields the coefficients for the error function $E(d)$, i.e.,

$$E(d) = c_1 d^2 + c_2 d + c_3.$$

The maximal expected variance of a point $\boldsymbol{q}$ in both x and y directions can thus be modelled by the multivariate normal distribution $\mathcal{N}(\mu, \Sigma)$, where:

$$\begin{aligned} \mu &= \boldsymbol{q}, \\ \Sigma &= \begin{bmatrix} E(||\boldsymbol{q}||_2)^2 & 0 \\ 0 & E(||\boldsymbol{q}||_2)^2 \end{bmatrix}. \end{aligned}$$

For each pair keypoint $\boldsymbol{p}^{\mathcal{U}} = [\boldsymbol{c}, \boldsymbol{o}]^T$, encoded in the unit space, the expected variance is modelled relative to the coordinate system origin for the point $\boldsymbol{c}$, and relative to $\boldsymbol{c}$ for the point $\boldsymbol{o}$. After applying the modelled multivariate normal distribution to each keypoint in the encoded constellation, we obtain a deformable constellation which is not affected by slight keypoint variations and is thus capable of addressing object deformations, as shown in Figure 4.7.

4.4 Constellation consolidation

A deformable keypoint constellation consists of a series of expected multivariate normal distributions of the encoded pair keypoints. All constellations, even though modelling different local object-structures, are encoded in the "same" unit space. Because of this, some distributions lie within other distributions, meaning that from a geometrical viewpoint they are very similar and can be merged, in order to reduce the total number of encoded distributions, as shown in Figure 4.8.

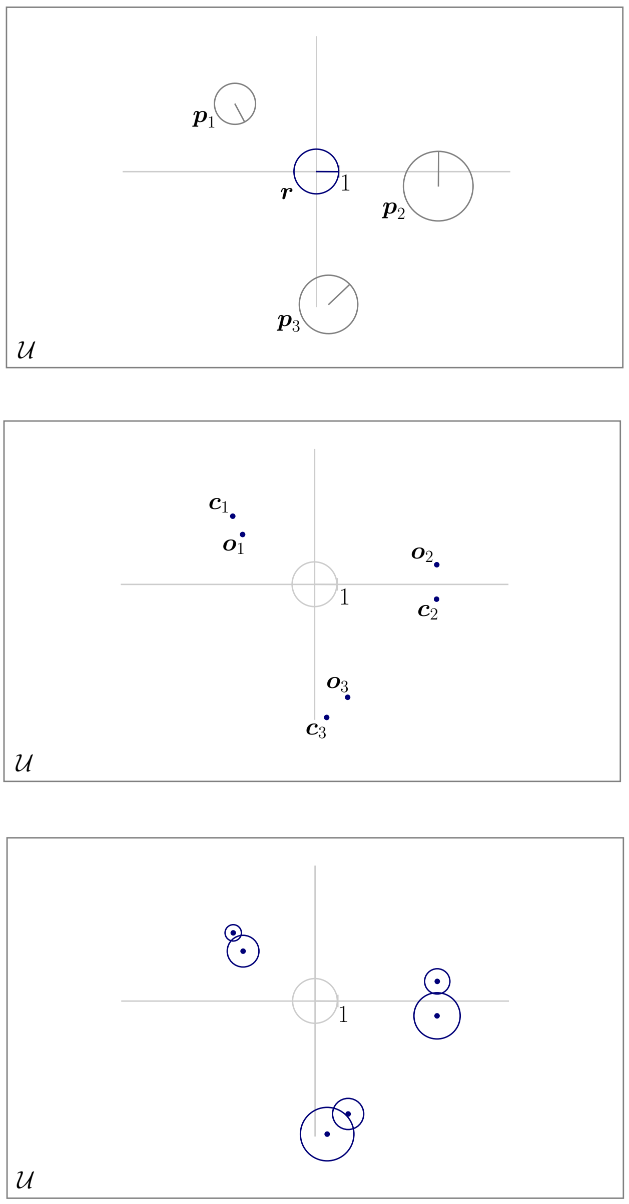


Figure 4.7: In the unit space, each keypoint is represented by two points, i.e., the center point $\boldsymbol{c}$ and the angular point $\boldsymbol{o}$. The expected variance of the keypoint center point and angular point is "attached" to each of these two points, forming a deformable keypoint constellation.

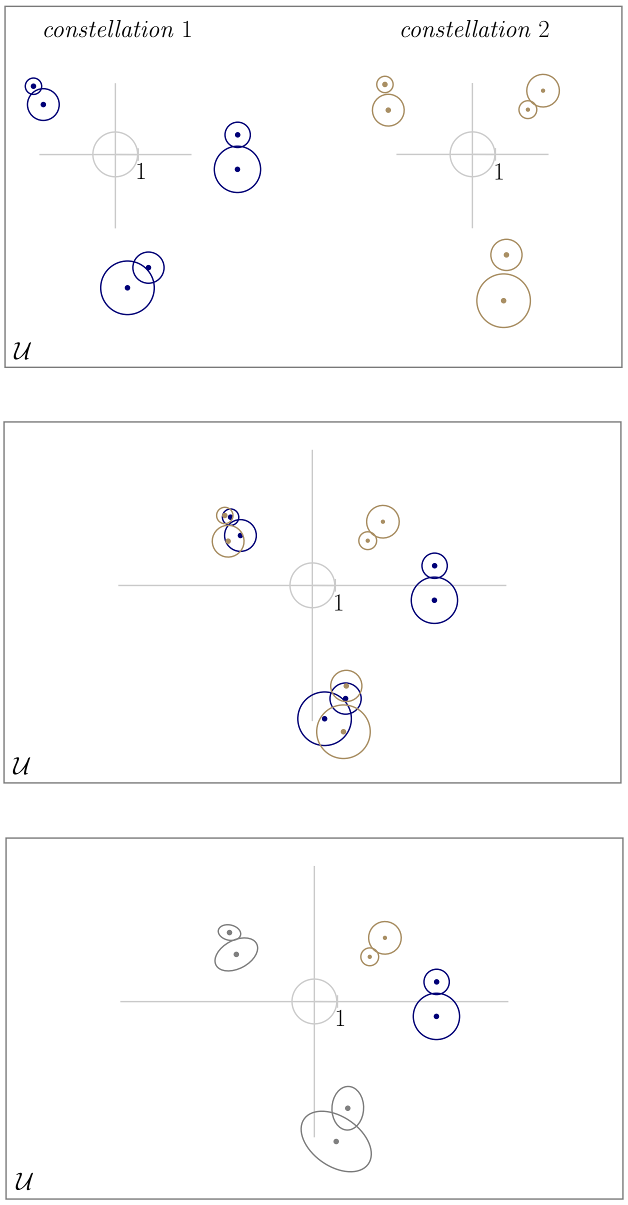


Figure 4.8: Certain distributions in the unit space overlap, meaning that from a geometrical viewpoint they are very similar. In order to reduce the total number of encoded distributions, overlapping distributions are merged together.

A distribution $\mathcal{N}_j$ lies within another distribution $\mathcal{N}_k$, if the mean $\mu_j \in \mathcal{N}_j$ lies within the distribution $\mathcal{N}_k$ under the confidence interval p. By computing the Mahalanobis radius of the ellipsoid $\Sigma_k \in \mathcal{N}_k$, which encloses the probability mass, we translate the confidence interval p into a Mahalanobis distance. The confidence interval Mahalanobis distance D_{conf} is computed using the inverse of the chi-square cumulative distribution function (cdf), with v degrees of freedom at the confidence value p,

$$D_{\text{conf}} = \text{Inv-}\chi^2(p, v).$$

Since Inv-χ^2 initially translates to the inverse of the Gamma cumulative distribution function with parameters $\alpha = \frac{v}{2}$ and $\beta = 2$, we can write:

$$D_{\text{conf}} = \text{Inv-Gamma}(p, \frac{v}{2}, 2).$$

The mean μ_j lies within the distribution $\mathcal{N}_k(\mu_k, \Sigma_k)$, if the Mahalanobis distance between μ_j and the distribution $\mathcal{N}_k$ is less than the computed confidence interval Mahalanobis distance D_{conf}:

$$\sqrt{(\mu_j - \mu_k)^T \cdot \Sigma_k^{-1} \cdot (\mu_j - \mu_k)} < D_{\text{conf}},$$

The consolidation of distributions is performed by means of the greedy approach. A distribution $\mathcal{N}_j$ that lies within two distributions $\mathcal{N}_a$ and $\mathcal{N}_b$, is merged with the distribution that contains the larger number of distributions lying within it, as illustrated in Figure 4.9. The mixture model is approximated by a single Gaussian by moment matching. A distribution consisting of n multivariate normal distributions is computed as follows,

$$\bar{\mu} = \sum_i^n \mu_i \cdot w_i,$$

$$\bar{\Sigma} = \Big(\sum_i^n (\Sigma_i + \mu_i \mu_i^T) \cdot w_i\Big) - \bar{\mu}\bar{\mu}^T.$$

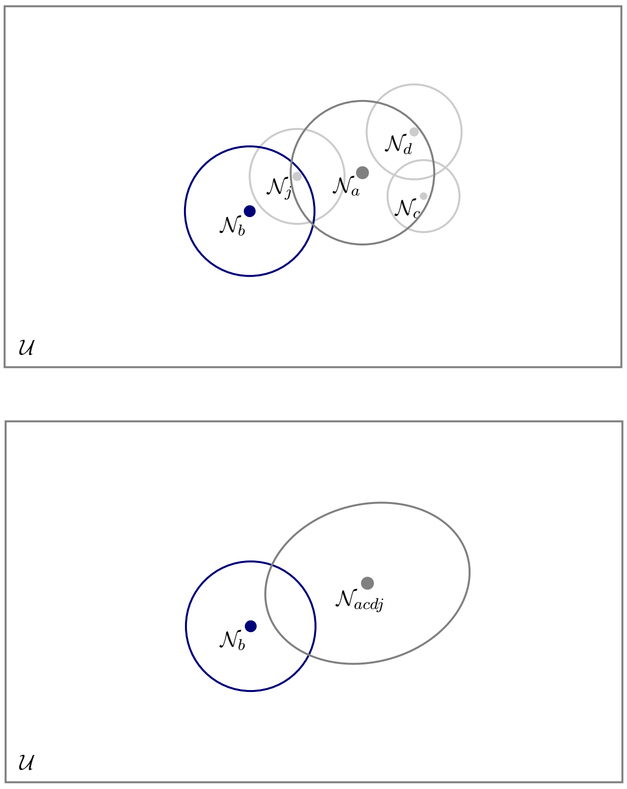


Figure 4.9: The merging approach symbolises a greedy approach, since in the illustrated case the distribution $\mathcal{N}_j$ is absorbed by the distribution $\mathcal{N}_a$, as $\mathcal{N}_a$ contains more distributions than $\mathcal{N}_b$.

4.5 The constellation object-model

The constellation model construction for a given object merely requires a single planar training image, i.e., *one-shot learning*, instead of a set of training images depicting the same object. This is due to the model encoding process, which takes keypoint variance into account and automatically constructs deformable constellations.

For an arbitrary object we thus construct an abstract, feature-based model representation $\mathcal{M}$, illustrated in Figure 4.10. The model consists of detected keypoints, their corresponding descriptors and detected constellations. Each keypoint points to multiple modelled variances of the neighbouring keypoints, i.e., points to a surrounding constellation. Additionally, the center of the object represented by the vector $\boldsymbol{w}$, the object bounding-box represented by the matrix $\boldsymbol{B}$ and an intensity template $I_{n\times n}$ of the image depicting the object are stored.

Given the illustrated model representation in Figure 4.10, it is easy to see the scaling potential of the constructed model with regards to a growing number of objects. Assume that we want to construct one constellation model for a number of objects. At first, for each object the keypoint constellations are encoded. Secondly, constellation consolidation is performed on all encoded constellations, since all constellations lie in the unit space. For each object, the center of the object represented by the vector $\boldsymbol{w}$, the object bounding-box represented by the matrix $\boldsymbol{B}$ and an intensity template $I_{n\times n}$ of the image depicting the object are stored. The so constructed model, consisting of multiple objects is illustrated in Figure 4.11.

Another aspect of the constellation model is the capability of incrementally adding a new object to an existing model. This merely requires adding new encoded keypoint constellations to the unit space of existing ones and consolidating the new constellations with existing ones. Additionally, the center of the new object, the object bounding-box and the object intensity template are added to the existing model.

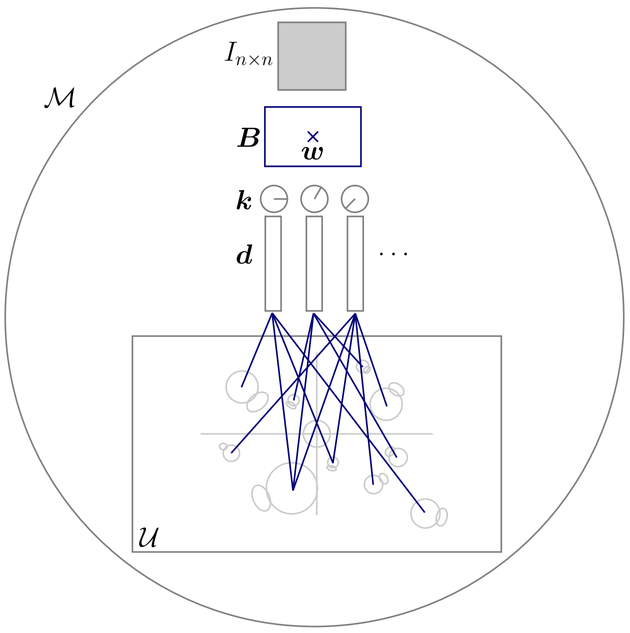


Figure 4.10: Model $\mathcal{M}$ of an encoded object. Each extracted keypoint $\boldsymbol{k}_i$ is represented by the corresponding descriptor $\boldsymbol{d}_i$, which points to one or more multivariate normal distributions encoded in the unit space, indicating the detected constellation around it.

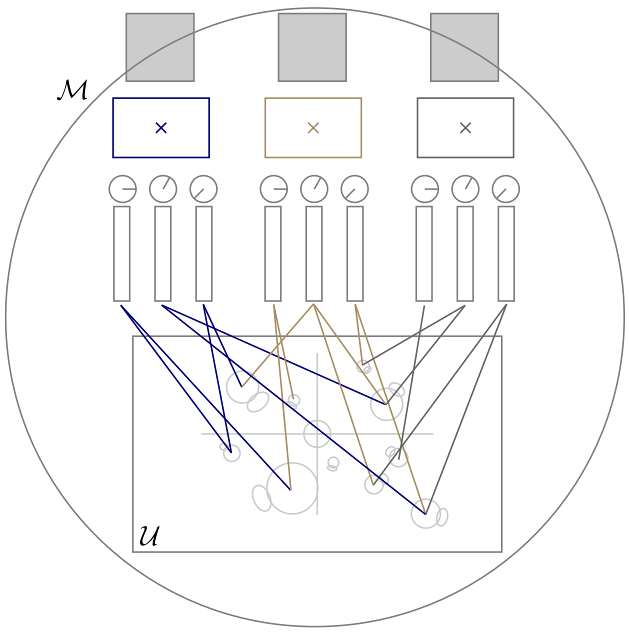


Figure 4.11: Model scaling. Representation for encoding a number of objects with one constellation model.

Chapter 5

Object detection by constellations

Object detection by constellation models proceeds as follows. From an arbitrary test image, containing or not containing the object of interest, image features are extracted. The extracted features are matched to the model features, linking the model of the object to the test image. For each matched feature, a constellation search is performed utilizing the encoded model with the aim to distinguish features located on the object from features located elsewhere. Feature votes for object center are accumulated, yielding potential object location hypothesis. For each hypothesis the object location in the test image is estimated using the matched features and the stored object bounding-box. The location estimation is refined and a final verification utilizing normalized cross-correlation between the estimated object location in the image and the stored intensity template is performed.

Section 5.1 describes the process of matching detected image features to model features, while a detailed description of filtering by constellation matching is given in Section 5.2. The process of object detection by features and detection verification is described in Section 5.3.

5.1 Feature matching

Extracted features are matched to features in the constructed model via the corresponding descriptors by nearest-neighbour matching. Each detected keypoint $\boldsymbol{k}_i$ is associated to the closes model keypoint $\boldsymbol{m}_i$ in terms of the smallest Euclidean distance between the descriptors $\boldsymbol{d}_i^k$ and $\boldsymbol{d}_i^m$, i.e.,

$$\boldsymbol{k}_i \rightarrow \boldsymbol{m}_i \quad \text{if} \quad \boldsymbol{d}_i^m = \arg\min_{\boldsymbol{d}_i^m}(||\boldsymbol{d}_i^k - \boldsymbol{d}_i^m||_2).$$

After detected keypoints have been matched to model keypoints, the encoded constellations in the model are utilized in order to suppress noisy keypoint matches.

5.2 Filtering by constellations

For each detected keypoint $\boldsymbol{k}_i$, associated with a model keypoint $\boldsymbol{m}_i$, a constellation search is performed. In the encoded model, every model keypoint $\boldsymbol{m}_i$ points to one or more expected variations of its neighbouring keypoints, encoded in the unit space as shown in Figure 4.10. It follows that for each keypoint match we obtain a subset of expected multivariate normal distributions, i.e., $\boldsymbol{k}_i \rightarrow \boldsymbol{m}_i \models \mathcal{N}'$ where $\mathcal{N}' \subset \mathcal{N}$. These serve for detecting neighbouring keypoints that form an encoded constellation with the current keypoint k_i as the root of the constellation. In general, the larger the number of keypoints fit in the encoded distributions, up to the size of the subset, the higher the probability that k_i is a keypoint located on the desired object.

Initially, all detected keypoints lie in the image space, i.e., $K \in \mathcal{I}$, all model keypoints in the model space, $M \in \mathcal{M}$, and all distributions lie in the unit space, i.e., $\mathcal{N} \in \mathcal{U}$. These need to be mapped from the image space to the unit space in order to verify whether any of the detected keypoints fall into the subset of distributions and form an encoded constellation with the keypoint $\boldsymbol{k}_i \in K$ as root. To address potentially stronger constellation deformations, the mapping from image space to unit space is done via the model space, as illustrated in Figure 5.1. At the initial step, the only information we can rely on is the association between the detected keypoint $\boldsymbol{k}_i$ and the model keypoint $\boldsymbol{m}_i$, as depicted by *1. iteration* in Figure 5.1. This is little information from which a similarity transformation $\boldsymbol{T}_{\mathcal{M}} : \boldsymbol{k}_i \rightarrow \boldsymbol{m}_i$ can be obtained, which maps all the detected keypoints in the image space to the model space. These are in turn mapped to the unit space via the transformation $\boldsymbol{T}_{\mathcal{U}} : \boldsymbol{k}_i^{\mathcal{M}} \rightarrow \boldsymbol{u}$, which is obtained by mapping the transformed root of the constellation $\boldsymbol{k}_i^{\mathcal{M}}$, which now lies in the model space, to the unit keypoint $\boldsymbol{u}$. In the unit space, the verification is performed whether any of the keypoints in the unit space $K^{\mathcal{U}}$, under the confidence interval p, fall into the subset of distributions $\mathcal{N}'$. Additionally, the transformation $\boldsymbol{M}_{\mathcal{U}} : \boldsymbol{m}_i \rightarrow \boldsymbol{u}$ transforms all model keypoints to the unit space, for which the same verification is performed. The verification itself is performed the same way as described in Section 4.4. In the case of strong constellation deformations, keypoints closer to the root of the constellation $\boldsymbol{k}_i$ will likely experience a smaller location variation opposed to keypoints that are further away. Since the model is constructed using a planar representation of the object, this means that keypoints which are further away from $\boldsymbol{k}_i$ will likely fall out of their designated distributions, in the event of a stronger object deformation. This problem can be addressed by refining the mapping step from image space to model space whenever a newly acquired information is available.

Let the detected keypoints $\boldsymbol{k}_j$, $\boldsymbol{k}_k$ and the model keypoints $\boldsymbol{m}_j$, $\boldsymbol{m}_k$ where $\boldsymbol{k}_j \rightarrow \boldsymbol{m}_j$ and $\boldsymbol{k}_k \rightarrow \boldsymbol{m}_k$, fall into the j-th and k-th closest distributions $\{\mathcal{N}_j, \mathcal{N}_k\} \in \mathcal{N}'$ under the previously computed transformations. This indicates that $\boldsymbol{k}_j$ and $\boldsymbol{k}_k$ fit in the encoded constellation around $\boldsymbol{k}_i$, and that the matches $\boldsymbol{k}_j \rightarrow \boldsymbol{m}_j$ and $\boldsymbol{k}_k \rightarrow \boldsymbol{m}_k$ are not noisy matches, since $\boldsymbol{m}_j$ and $\boldsymbol{m}_k$ fall into the same distribution. This represents new information that can be used to compute a more accurate transformation in the mapping from the image space to the model space, as depicted by *2. iteration* in Figure 5.1. By repeating the mapping step, the transformation $\boldsymbol{T}_{\mathcal{M}} : \{\boldsymbol{k}_i, \boldsymbol{k}_j, \boldsymbol{k}_k\} \rightarrow \{\boldsymbol{m}_i, \boldsymbol{m}_j, \boldsymbol{m}_k\}$ maps all keypoints detected in the image space to the model space with a greater accuracy. This process is repeated until the set of associated key-

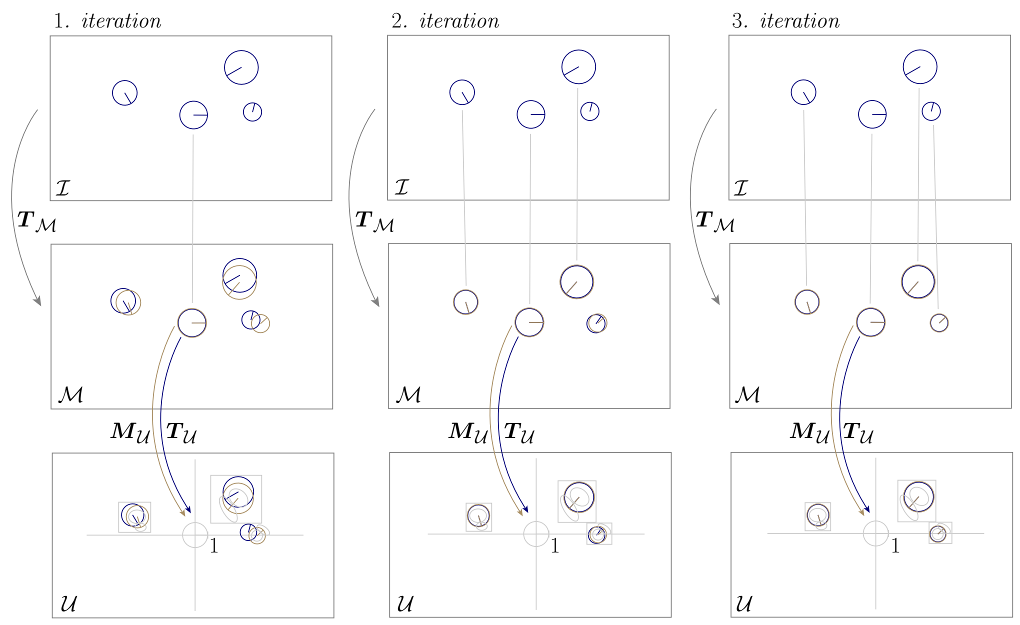

1. *iteration*
2. *iteration*
3. *iteration*
$\mathcal{I}$
$\boldsymbol{T}_{\mathcal{M}}$
$\mathcal{M}$
$\boldsymbol{M}_{\mathcal{U}}$
$\boldsymbol{T}_{\mathcal{U}}$
$\mathcal{U}$
1

Figure 5.1: In the first iteration, the transformation $\boldsymbol{T}_{\mathcal{M}}$ is estimated considering only one keypoint correspondence, depicted by a gray line. If, under the constraints of the encoded distributions, neighbouring keypoints are found, these are considered as new correspondence is the next iteration, from which a new transformation $\boldsymbol{T}_{\mathcal{M}}$ is estimated.

points in the constellation rooted at $\boldsymbol{k}_i$ stops changing, as depicted by *3. iteration* in Figure 5.1. Essentially we search for the number of neighbouring keypoints that $\boldsymbol{k}_i$ can "grasp on", within the limitation of the encoded distributions. The higher the number of keypoints that form a constellation around $\boldsymbol{k}_i$, the higher the probability that $\boldsymbol{k}_i$ belongs to the desired object. Keypoints, for which no neighbouring keypoints where found to fit an assigned constellation are rejected from the initial set of associated keypoints. This process in fact acts as a filter, filtering out uncertain keypoint matches. A brief outline of filtering keypoints by constellation matching is given in Algorithm 1.

Algorithm 1 Filtering keypoints by constellation matching

```
 1: for ∀ $\boldsymbol{k}_i \in K$ do
 2:     $\boldsymbol{m}_i \leftarrow$ model keypoint associated with $\boldsymbol{k}_i$
 3:     $\mathcal{N} \leftarrow$ distributions associated with $\boldsymbol{k}_i$
 4:     $\{\cdot\}_K \leftarrow$ add $\boldsymbol{k}_i$ to set
 5:     $\{\cdot\}_M \leftarrow$ add $\boldsymbol{m}_i$ to set
 6:     repeat
 7:         $\boldsymbol{T}_{\mathcal{M}} : \{\cdot\}_K \rightarrow \{\cdot\}_M$
 8:         $K^{\mathcal{M}} \leftarrow K \cdot \boldsymbol{T}_{\mathcal{M}}$
 9:         $\boldsymbol{T}_{\mathcal{U}} : \boldsymbol{k}_i^{\mathcal{M}} \rightarrow \boldsymbol{u}$
10:         $\boldsymbol{M}_{\mathcal{U}} : \boldsymbol{m}_i \rightarrow \boldsymbol{u}$
11:         $K^{\mathcal{U}} \leftarrow K^{\mathcal{M}} \cdot \boldsymbol{T}_{\mathcal{U}}$
12:         $M^{\mathcal{U}} \leftarrow M \cdot \boldsymbol{M}_{\mathcal{U}}$
13:         if $\exists\, \boldsymbol{k}_j^{\mathcal{U}} \in K^{\mathcal{U}}$ such that $\boldsymbol{k}_j^{\mathcal{U}} \in \mathcal{N}$ then
14:             $\{\cdot\}_K \leftarrow$ add $\boldsymbol{k}_j$ to set
15:             $\{\cdot\}_M \leftarrow$ add $\boldsymbol{m}_j$ to set
16:         end if
17:     until size of $\{\cdot\}_K$ stops changing
18:     if $|\{\cdot\}_K| = 1$ then
19:         remove $\boldsymbol{k}_i$ from $K$
20:     end if
21: end for
```

5.3 Object detection by features

After the constellation filtering, the remaining keypoints are used for object localization. Object localization is performed by the Generalised Hough Transform [32]. In general, each keypoint casts a vote for which a cell in the accumulator array is incremented. The cells with the highest number of accumulated votes indicate potential object locations in the test image. Each potential location, i.e., location hypothesis, is verified by bounding the hypothetical location of the object in the test image, as described in Section 5.3.1 and performing a final detection verification step, described in Section 5.3.2.

5.3.1 Object localization

In the GHT accumulator array, each cell $h_i \in A$ that exceeds a threshold τ_{GHT} represents a potential object location in the test image. The threshold serves to eliminate cells with insufficient number of votes and are less likely to correspond to the object. Thresholding the accumulator array thus yields a set of object location hypothesis, i.e., $H = \{h_i \in A \mid h_i > \tau_{\text{GHT}} \; ; \; i = 1, ..., N\}$. Each location hypothesis in turn yields a set of keypoints supporting that hypothesis, i.e., $\forall \; h_i \in H \models K$, where $K = \{\boldsymbol{k}_i \in \mathcal{I} \mid \boldsymbol{k}_i \in h_i\}$. For a given hypothesis h_i supported by a set of keypoints K, the location of the object in the test image is estimated by considering the links between detected and model keypoints. Since each detected keypoint in K is associated to a model keypoint, i.e., $\forall \, \boldsymbol{k}_i \in K \models \boldsymbol{k}_i \to \boldsymbol{m}_i$, K yields the set of model keypoints $M = \{\boldsymbol{m}_i \in \mathcal{M} \mid \boldsymbol{m}_i \to \boldsymbol{k}_i\}$. Thus for a hypothesis h_i, K and M represent the set of matched keypoints which map the object model to a specific location in the test image. The refined object location for a hypothesis h_i can be determined by computing the transformation parameters which map the set of model keypoints to the set of detected keypoints, i.e., $\boldsymbol{T} : M \to K$. However, simply using all associated keypoints to compute the transformation is prone to errors due to potential outliers as discussed in Section 3.4. Therefore, object localization is performed in two stages: (i) filtering keypoints by their voting scale and (ii) iterative bounding-box estimation. A brief outline of the object localization procedure is given in Algorithm 2.

Algorithm 2 Object detection

```
1: for $h_i \in H$ do
2:     $K \leftarrow$ keypoints supporting $h_i$
3:     procedure Object localization
4:         $K' \leftarrow$ keypoints in $K$ filtered by scale
5:         do Iterative bounding-box estimation
6:     end procedure
7:     do Normalized cross-correlation verification
8: end for
```

Each keypoint that casted a vote in the accumulator array voted for a certain scale of the object. Outlier influence on the scale of the object is omitted by sorting keypoints by their voting scale and removing α percent of the keypoints that correspond to the smallest and largest voting scales. The voting scale of a keypoint is determined from the transformation $\boldsymbol{T} : \boldsymbol{m}_i \rightarrow \boldsymbol{k}_i$, by computing the Euclidean norm between the first two diagonal elements of the transformation matrix, i.e., $S = ||\boldsymbol{T}_{1,1} - \boldsymbol{T}_{2,2}||_2$.

Keypoints that withstood scale filtering are used to determine the transformation parameters which map the object model to a location in the test image. The correspondences between model and detected keypoints are processed by MLESAC [34], a method for robustly estimating multiple view relations from point correspondences. The result is a subset of correspondences, i.e., $M' \subseteq M$ and $K' \subseteq K$, from which the best transformation parameters can be estimated, resulting in a transformation $\boldsymbol{T} : M' \rightarrow K'$ which maps the object model to a location in the image. Using this transformation the object bounding-box $\boldsymbol{B}$ is mapped to the test image, bounding a fragment of the test image on which the object of interest is hypothetically located. A good practice is to check for possible skewing of the transformed bounding-box $\boldsymbol{B}'$, check whether $\boldsymbol{B}'$ falls within the test image borders, and whether the point $\boldsymbol{w}'$, transformed by $\boldsymbol{T}$, falls within $\boldsymbol{B}'$. If these requirements are not met, the set of keypoints is considered to be a noisy set, meaning that the hypothesis is rejected and a new hypothesis is verified. Additional steps toward refining the mapped bounding-box $\boldsymbol{B}'$ include verifying whether all the detected keypoints in the set K' are located within $\boldsymbol{B}'$, which is a reasonable assumption since the bounding-box bounds the object and all keypoints located on the object. If a detected keypoint $\hat{\boldsymbol{k}}_i \in K'$ lies outside of $\boldsymbol{B}'$, then this keypoint is considered not to be located on the object. The detected keypoint $\hat{\boldsymbol{k}}_i$ and the associated model keypoint $\hat{\boldsymbol{m}}_i$ are removed from the sets K and M, yielding new subsets $K'' \subset K'$ and $M'' \subset M'$. From these matches a new transformation $\boldsymbol{T} : M'' \to K''$ is computed which maps the object bounding-box $\boldsymbol{B}$ to a more refined bounding-box $\boldsymbol{B}'$ on the test image. The whole process, dubbed *iterative bounding-box estimation* and illustrated by pseudo Algorithm 3, is repeated until the set of keypoints located within $\boldsymbol{B}'$ stops changing.

Algorithm 3 Iterative bounding-box estimation

```
 1: $M \leftarrow$ set of model keypoints
 2: $K \leftarrow$ set of detected keypoints
 3: **repeat**
 4:     $M', K' \leftarrow \text{MLESAC}(M, K)$
 5:     $\boldsymbol{T} : M' \to K'$
 6:     $\boldsymbol{B}' \leftarrow \boldsymbol{B} \cdot T$
 7:     $\boldsymbol{w}' \leftarrow \boldsymbol{w} \cdot T$
 8:     **if** $\boldsymbol{B}'$ strongly skewed **then**
 9:         break
10:     **end if**
11:     **if** $\boldsymbol{w}'$ not within $\boldsymbol{B}'$ **then**
12:         break
13:     **end if**
14:     **if** $\exists\, \boldsymbol{k}_i \in K$ such that $\boldsymbol{k}_i \notin \boldsymbol{B}'$ **then**
15:         $K \leftarrow \{\boldsymbol{k}_i \mid \boldsymbol{k}_i \in K \ ; \ \boldsymbol{k}_i \in \boldsymbol{B}'\}$
16:         $M \leftarrow \{\boldsymbol{m}_i \mid \boldsymbol{m}_i \in M \ ; \ \boldsymbol{m}_i \to \boldsymbol{k}_i\}$
17:     **end if**
18: **until** size of $K$ stops changing
```

5.3.2 Detection verification

The final estimated bounding-box $\boldsymbol{B}'$ bounds a fragment of the test image on which the object of interest is hypothetically located. A final verification step is undertaken in order to verify that the detected region corresponds to the object of interest. The detected region in the test image is transformed to an intensity template J of size $n \times n$ pixels. The detected object intensity template J and the model object intensity template I are compared using the normalized cross-correlation.

The normalized cross-correlation is used since the brightness of the bound fragment on the test image can vary due to lighting and exposure conditions, compared to the object template image. The normalization is performed by subtracting the mean value of each template and dividing by the standard deviation of both templates, i.e.,

$$\begin{aligned} \mathrm{ncc}(J, I) &= \frac{1}{n} \sum_{x,y}^{n} \frac{(J(x,y) - \bar{J})(I(x,y) - \bar{I})}{\sigma_J \sigma_I}, \\ \bar{J} &= \frac{1}{n} \sum_{x,y}^{n} J(x,y), \\ \sigma_J &= \sqrt{\frac{1}{n} \sum_{x,y}^{n} (J(x,y) - \bar{J})^2}, \end{aligned}$$

where n denotes the number of pixels in J, $\bar{J}$ the mean value of J and σ_J the standard deviation of J. If the final matching score exceeds a threshold τ_{ncc}, the object location hypothesis passes the verification test yielding a detected object in the test image, otherwise the hypothesis is rejected.

Chapter 6

Experimental evaluation

The aim of the proposed keypoint constellation model is that, for a given object and test image, filter out detected features in the test image that do not correspond to features located on the object of interest. The proposed constellation model is designed to merely enhance basic feature-based object detection methods, since constellations are computed and utilized ”on top” of extracted features. Considering that the proposed similarity threshold in [11] does not always prove efficient, the experimental evaluation consists of comparing the detection performance of six different models, summarized in Table 6.1.

6.1 Implementation details

The experimental evaluation is conducted as follows. For each object class in the dataset, a model is constructed from a single planar training image. For a given object class and a given test image, the object location in the image is determined by utilizing the corresponding model. Features are extracted from the test image and matched to features in the model representation. Non-discriminative features are filtered out by six different approaches. Features that withstood the filtering process are used to determine the object location in the test image in a cascade of keypoint filtering, iterative bounding-box estimation and detection verification steps. The following variations of feature-based object models where tested:

$\mathcal{B}$ denoting a model consisting of SIFT features and the object detection algorithm without the use of the similarity threshold.

$\mathcal{B}_\tau$ denoting a model consisting of SIFT features and the object detection algorithm with the use of the similarity threshold.

$\mathcal{C}^E$ denoting a model consisting of SIFT features, encoded constellations with empirically modelled variance and the object detection algorithm with constellation filtering and without the use of the similarity threshold.

$\mathcal{C}^E_\tau$ a model consisting of SIFT features, encoded constellations with empirically modelled variance and the object detection algorithm with constellation filtering and with the use of the similarity threshold.

$\mathcal{C}^N$ denoting a model consisting of SIFT features, encoded constellations with numerically modelled variance and the object detection algorithm with constellation filtering and without the use of the similarity threshold.

$\mathcal{C}^N_\tau$ denoting a model consisting of SIFT features, encoded constellations with numerically modelled variance and the object detection algorithm with constellation filtering and with the use of the similarity threshold.

For convenience, the tested models are summarized in Table 6.1.

6.1.1 Parameters

All parameters used in the experimental evaluation are listed in Table 6.2 for convenience. The SIFT algorithm parameters are constant throughout the experiment. The implementation used is the evaluation is the OpenCV [1]

[1]`http://opencv.org/`

Table 6.1: Tested models.

Model notation	Similarity threshold	Empiric variance	Numeric variance
$\mathcal{B}$	$\times$	$\times$	$\times$
$\mathcal{B}_\tau$	✓	$\times$	$\times$
$\mathcal{C}^E$	$\times$	✓	$\times$
$\mathcal{C}^E_\tau$	✓	✓	$\times$
$\mathcal{C}^N$	$\times$	$\times$	✓
$\mathcal{C}^N_\tau$	✓	$\times$	✓

based VLFeat [2] implementation of the SIFT. During the model construction process, training images are resized to a fixed size by setting the longest axis to 300 pixels, i.e., $I_{\text{max}} = 300$. The accumulator array is set to a fixed size by setting the longest side of the accumulator to $A_{\text{max}} = 50$ pixels, relative to the longest side of the test image. All accumulator values are normalized and the threshold τ_{GHT} is set to 0.1, meaning that cells which accumulated less than 10% of the highest scoring cell are suppressed. During the model construction process the value of the τ_{dst} parameter, in the ϵ region determination around a given keypoint, is set to 0.02. During model construction, the multivariate normal distribution confidence interval for the constellation consolidation process is set to 0.68, i.e., $\sigma \approx 1$. The confidence interval for the detection algorithm during keypoint filtering by constellation matching is set to 0.98, i.e., $\sigma \approx 2.5$.

6.2 Performance evaluation

The performance of the models was evaluated on the challenging real-world dataset FlickrLogos-32, described in Section 6.2.1. A description of the performance measures used in the evaluation is given in Section 6.2.2.

[2]`http://www.vlfeat.org/`

Table 6.2: Experimental evaluation parameters.

Name	Value	Description
$\tau_{extrema}$	0.03	SIFT extrema point threshold
$FirstOctave$	-1	SIFT index of the first octave of the DoG scale space
$PeakThresh$	5	SIFT peak selection threshold
$EdgeThresh$	10	SIFT non-edge selection threshold
$NormThresh$	0	SIFT minimum descriptor $L2$-norm before normalization
n	4	MLESAC minimum number of required data points
w	0.4	MLESAC fraction of correctly matched data points
α	0.95	MLESAC probability of inliers
N	100	MLESAC approximate number of required iterations
$\tau_{inliers}$	1	MLESAC inlier error threshold
I_{max}	300 pixels	Model construction training image maximal axis size
τ_{dst}	0.02	Model construction ϵ region determination value
p_1	0.68 ($\sigma \approx 1$)	Model construction MND confidence interval
p_2	0.98 ($\sigma \approx 2.5$)	Detection algorithm MND confidence interval
A_{max}	50 pixels	GHT accumulator array maximal axis size
τ_{GHT}	0.1	GHT accumulator array threshold
τ_{ncc}	0.5	Normalized cross-correlation threshold
τ_{pvoc}	0.5	Pascal VOC overlap threshold

6.2.1 The FlickrLogos-32 dataset

In order to assure a realistic evaluation of object detection methods, authors in [28] published a large dataset [3], dubbed *FlickrLogos-32*, depicting logotypes in real-world environments. A brief visual summary of the dataset is depicted in Figure 6.1. The challenging dataset consists of 32 classes of logotypes obtained from the Flickr website, whereby all logotypes have an approximately planar surface. For each logotype class, 70 images containing at least one instance of the class are available. The whole dataset thus consists of 2240 logotype images, with the maximal image axis fixed at 1024 pixels. The 32 logotype classes are: Adidas, Aldi, Apple, Becks, BMW, Carlsberg, Chimay, Coca-Cola, Corona, DHL, Esso, Erdinger, Fedex, Ferrari, Ford, Fos-

[3] `http://www.multimedia-computing.de/flickrlogos`

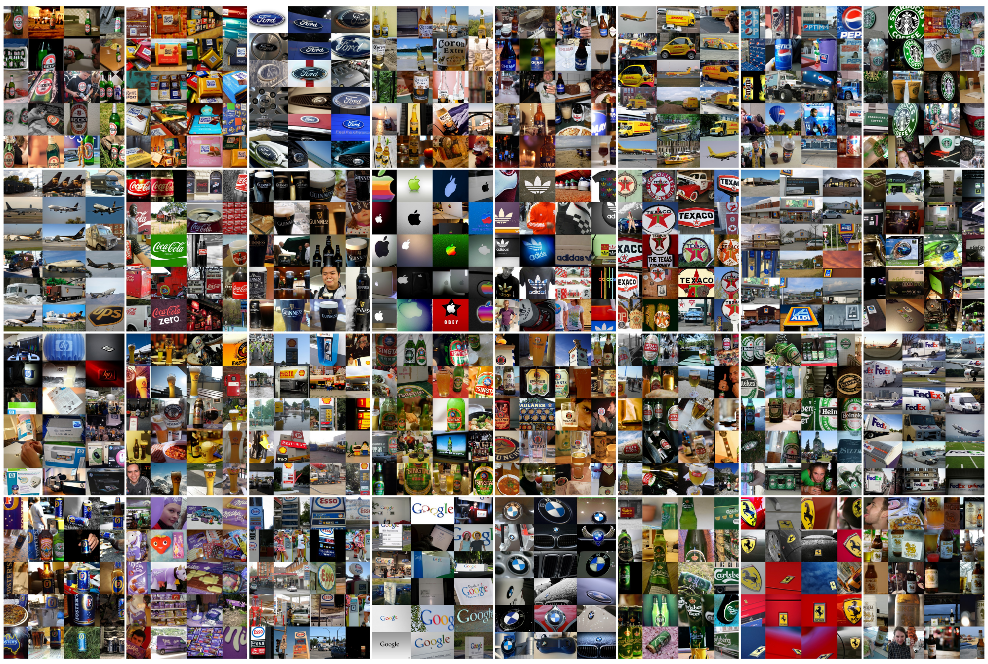


Figure 6.1: Visual summary of all 32 classes, as provided by [28].

ter's, Google, Guiness, Heineken, HP, Milka, Nvidia, Paulaner, Pepsi, Ritter Sport, Shell, Singha, Starbucks, Stella Artois, Texaco, Tsingtao and UPS. Additionally, a set of images containing no logotype class is provided. However, this set is left out from the experimental evaluation since our aim is to verify the affect of constellations of keypoints on object detection in terms of noise reduction and detection performance.

Although the dataset contains depictions of logotypes, these can be considered as rigid $2D$ objects with an approximately planar surface. The challenge in the dataset arises from the great variance of object sizes, from tiny logos in the background to image-filling views, perspective deformation and images containing multiple object instances. Given the good quality of the images in the dataset from which a sufficient amount of information can be extracted, the dataset is suitable for the evaluation of keypoint based object detection methods.

6.2.2 Performance measures

Detection performance of the object detection algorithm utilizing a given model is evaluated based on the overlap area of two annotations, i.e., D, denoting the detected region in a test image, and G, denoting the ground-truth annotation in the dataset. Detections are verified using the Pascal VOC overlap [35] measure, defined as the fraction of the intersection between two bounding boxes and their union, i.e.,

$$score = \frac{area(D \cap G)}{area(D \cup G)}.$$

If the computed score exceeds the threshold $\tau_{\text{pvoc}} = 0.5$, the detection of the object is considered to be successful, otherwise the detection is considered to be unsuccessful. The performance is evaluated based on the precision, recall and F-measure values, where:

- *Precision* denotes the fraction of retrieved instances that are relevant,
- *Recall* denotes the fraction of relevant instances that are retrieved,
- *F-measure* denotes the harmonic mean of precision and recall, i.e.,

$$F\text{-measure} = 2\,\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

6.3 Results

Figure 6.2 depicts the total number of generated hypothesis, i.e., H_{total} and the number of hypothesis exceeding the threshold τ_{GHT}, i.e., $H_{\tau_{GHT}}$, per tested model. The proposed constellation models $\mathcal{C}^E$, $\mathcal{C}^E_\tau$, $\mathcal{C}^N$ and $\mathcal{C}^N_\tau$ significantly reduce H_{total} and $H_{\tau_{GHT}}$ when compared to the basic models $\mathcal{B}$ and $\mathcal{B}_\tau$. The same trend is visible in Figure 6.3, which depicts the average number of generated hypothesis, i.e., μ_{total} and the average number of hypothesis exceeding the threshold τ_{GHT}, i.e., $\mu_{\tau_{GHT}}$.

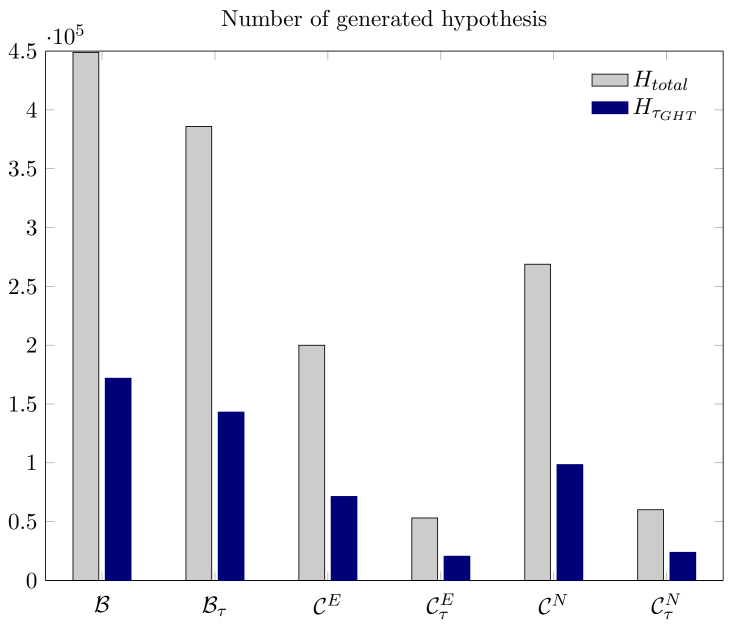


Figure 6.2: The total number of generated object location hypothesis H_{total} vs. the number of generated hypothesis exceeding the threshold τ_{GHT}, i.e., $H_{\tau_{GHT}}$. The proposed constellation models exhibit a significantly smaller number of generated hypothesis and hypothesis which need to be verified in case with and without the similarity threshold.

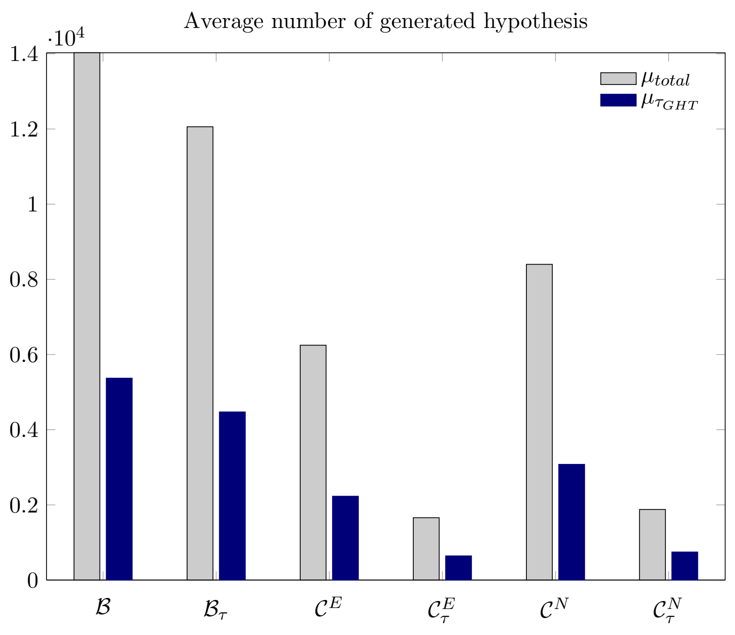


Figure 6.3: The mean of the total number of generated object location hypothesis μ_{total} vs. the mean of the number of generated hypothesis exceeding the threshold τ_{GHT}, i.e., $\mu_{\tau_{GHT}}$. The proposed constellation models exhibit a significantly smaller average number of generated hypothesis and hypothesis which need to be verified in case with and without the similarity threshold.

Table 6.3 gives H_{total} and $H_{\tau_{GHT}}$ per dataset class. In general, the proposed constellation models produce a significantly lower number of hypothesis which need to be verified. A significant decrease in the number of generated hypothesis is visible for the object class *apple*. This is because the apple logotype is a textureless object, meaning that very little, if any, features that describe the object can be extracted. The generated hypothesis are thus mostly noisy object location hypothesis. However, for logotypes from which enough information was extracted, such as *ups*, H_{total} is significantly reduced, e.g., for the given object class from 15407 to 3552 and in the case of $H_{\tau_{GHT}}$ from 6254 to 1865. In general, this trend is visible throughout all object classes.

Although the proposed constellation models significantly reduce the number of hypothesis H_{total} and $H_{\tau_{GHT}}$, the occurring question is how detection performance is affected. Figure 6.4 depicts the Precision-Recall curves for the tested models. In general, there is no noticeable difference in the performance of the models, implying that the proposed constellation models do not affect detection performance significantly. This in turn implies that the proposed constellation models merely reduce the number of noisy object location hypothesis. The same trend is visible in Figure 6.5, depicting the F-measure for each model with respect to the varying threshold τ_{ncc}.

Detection performance for each object class with respect to each tested model is given in Table 6.4. For most object classes, poor performance is visible with precision and recall values equal to zero, meaning that no instances of those objects were detected. In general this has to do with the strong intra-object-class variations of the logotype classes, which in some cases represent extremely hard detection tasks. Some of these challenging variations can be viewed in the visual summary of the dataset in Figure 6.1. Given that our model is trained from a single planar training image, for a given object class, this additionally increases the difficulty of distinguishing between stronger intra-object-class variations. However, in general the performance between models is more or less coherent, as visible Figure 6.4. Although cases where the basic model slightly outperforms the constellation models can be found for objects such as *aldi* and *google*, cases where the constellation model outperforms the basic model can also be found for objects such as *milka* and *dhl*.

Figure 6.6 depicts an example of the detection performance for all tested models for a test image of the object class *milka*. The constellation models potential, in the case of $\mathcal{C}^E$, is clearly visible, since it is able to detect logotypes in the upper left, and bottom right corners where the basic model fails, due to to presence of significant noise in the case of $\mathcal{B}$, and due to lack of information in the case of $\mathcal{B}_\tau$. A similar trend is depicted in Figure 6.7 and Figure 6.8 for the object classes *shell* and *esso*. The basic model is unable to detect all logotypes, due to presence of significant noise in the case of $\mathcal{B}$, and due to lack of information in the case of $\mathcal{B}_\tau$. The constellation models $\mathcal{C}^E$ and $\mathcal{C}^N$, however, are able to detect all logotypes. The filtering capabilities of the constellation models are depicted in Figure 6.9 and Figure 6.10. In both cases, the constellation model filters out a significant amount of keypoints, which are not located on the object of interest, and reduces the presence of noise in the accumulator array. A slight improvement in the detection accuracy is also visible, where the bounding-boxes, which bound the object of interest are more refined in the case of the constellation model.

A drawback of the constellation model is a degraded performance in cases of significant object deformations. In contrast to the basic feature model, the allowed deformation of the constellation model is restricted by the encoded geometry, whereas the unconstrained basic model allows for a higher degree of deformation. These shortcomings can be observed in Figure 6.11 and Figure 6.12. In the first case, two *dhl* logotypes are present in the test image. The logotype on the front of the delivery truck is slightly deformed, whereas the logotype on the side is subjected to a higher degree of deformation. The basic models are more or less able to detect both logotypes, whereas the constellation models fail to detect the logotype, located on the side of the delivery truck. A similar case is visible in Figure 6.12, depicting the *ford*

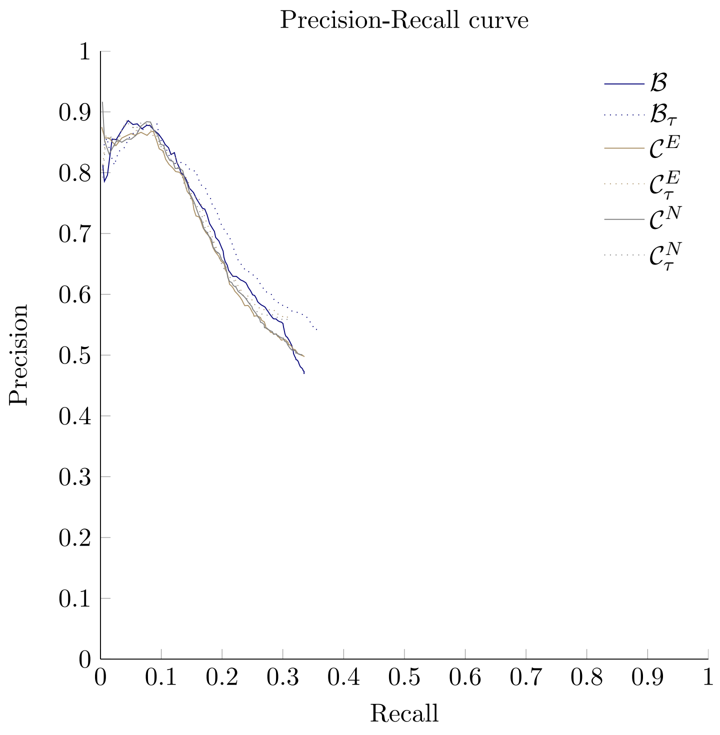


Figure 6.4: Performance evaluation of the tested models in terms of the Precision-Recall curve. The performance of the models does not differ significantly, meaning that the proposed constellation models do not affect detection performance significantly.

logotype at a stronger deformation. In both cases, the constellation models fail to detect objects at large perspective deformations. This is due to the encoded geometry restrictions, which do not allow for large deformations of keypoint locations, other than those that where modelled in process of model construction.

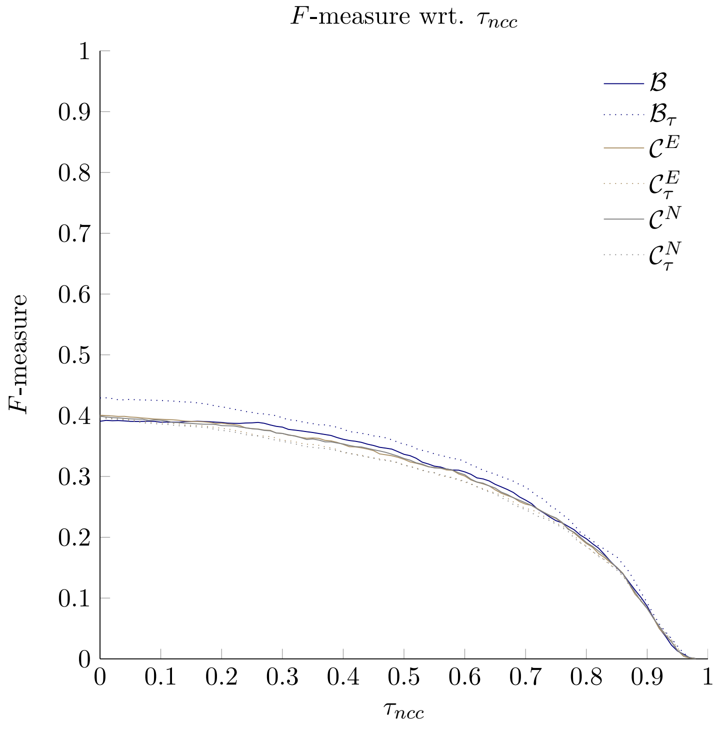


Figure 6.5: F-measure with respect to the normalized cross-correlation threshold τ_{ncc}. The F-measure of the tested models does not differ significantly, implying that the proposed constellation models do not affect detection performance significantly.

Table 6.3: Number of generated object location hypothesis. H_{total} denotes the total number of generated hypothesis and $H_{\tau_{GHT}}$ the number of generated hypothesis exceeding τ_{GHT}. Shaded columns represent tested models without the use of the similarity threshold.

	H_{total}						$H_{\tau_{GHT}}$					
	$\mathcal{B}$	$\mathcal{B}_\tau$	$\mathcal{C}^E$	$\mathcal{C}^E_\tau$	$\mathcal{C}^N$	$\mathcal{C}^N_\tau$	$\mathcal{B}$	$\mathcal{B}_\tau$	$\mathcal{C}^E$	$\mathcal{C}^E_\tau$	$\mathcal{C}^N$	$\mathcal{C}^N_\tau$
adidas	15316	11353	6189	1530	8357	1781	9064	8445	4750	1258	6100	1500
aldi	15297	13962	5601	1176	9212	1669	5545	4805	1992	570	3375	786
apple	7204	3866	84	8	1861	76	5377	3776	84	8	1861	76
becks	15186	13740	7675	2216	9972	2617	4572	3002	1429	722	2052	768
bmw	13773	6337	2318	871	3603	847	8862	5090	2051	773	3161	758
carlsberg	13524	14075	8074	2054	11330	2618	4821	3447	2048	493	3071	660
chimay	13986	12488	7951	2131	10236	2300	4661	2608	1409	468	1767	539
cocacola	16863	11120	3923	1253	5211	1272	8849	4972	2203	774	2765	788
corona	15610	12692	7655	2608	10323	2958	4268	2685	1449	795	2002	885
dhl	13534	11842	7745	962	10365	1102	6230	6574	4406	729	5950	821
erdinger	14110	13436	9385	3803	9962	3361	3381	2072	1351	897	1468	860
esso	11849	14855	7178	1246	10470	1210	4828	6682	2802	628	3624	629
fedex	17253	7974	4158	1459	5869	1466	9092	3465	1859	740	2478	724
ferrari	14141	7688	2611	1272	3218	1383	5070	3728	1526	822	1688	828
ford	12682	8623	2986	737	4316	903	5040	4004	2012	469	2864	534
fosters	13071	14074	10069	1716	11780	1948	5915	6076	3087	1002	3540	1133
google	13089	8691	4112	1255	5528	1348	5094	3793	2086	596	2948	610
guiness	15330	12120	4678	1589	6974	1823	3872	3483	1715	615	2346	846
heineken	14787	15821	10330	3197	12066	3674	5040	3262	1639	654	2049	647
hp	10997	8456	3074	700	5503	778	3960	4663	2117	528	3371	567
milka	12569	13740	6493	1611	10657	2552	5397	5238	3087	908	4997	1383
nvidia	10608	10377	6148	239	8929	501	5130	7273	3875	166	4737	376
paulaner	13562	14377	9090	2776	10658	3020	2760	1746	930	465	1023	509
pepsi	15545	15869	7545	577	9863	627	8718	13892	6634	465	9105	498
rittersport	15479	14332	9039	3657	11226	3755	3991	2570	1506	681	1846	699
shell	11962	13109	5541	1255	8988	2050	4812	6475	2769	1040	4385	1692
singha	14240	14763	6913	1593	9613	1988	5335	4426	2570	731	3327	881
starbucks	15405	12868	6020	2278	7646	2343	1143	725	438	272	534	307
stellaartois	15631	13787	7750	2855	9448	3024	3241	1364	810	580	1143	607
texaco	15161	13617	7856	2194	9548	2344	6150	4542	2428	878	3007	912
tsingtao	15736	13774	8182	1772	9806	1981	5409	3567	2468	558	2911	598
ups	15407	12086	3552	520	6258	805	6254	4680	1865	329	3024	483
Σ	448907	385912	199925	53110	268796	60124	171881	143130	71395	20614	98519	23904
μ	14028	12060	6248	1660	8400	1879	5371	4473	2231	644	3079	747
σ	2015	2898	2499	935	2728	954	1871	2450	1293	255	1710	326

Table 6.4: Detection performance for each object class in terms of precision, recall and F-measure. Shaded columns represent tested models without the use of the similarity threshold.

	Precision						Recall						F-score					
	$\mathcal{B}$	$\mathcal{B}_\tau$	$\mathcal{C}^E$	$\mathcal{C}^E_\tau$	$\mathcal{C}^N$	$\mathcal{C}^N_\tau$	$\mathcal{B}$	$\mathcal{B}_\tau$	$\mathcal{C}^E$	$\mathcal{C}^E_\tau$	$\mathcal{C}^N$	$\mathcal{C}^N_\tau$	$\mathcal{B}$	$\mathcal{B}_\tau$	$\mathcal{C}^E$	$\mathcal{C}^E_\tau$	$\mathcal{C}^N$	$\mathcal{C}^N_\tau$
adidas	0.000	0.000	0.032	0.000	0.061	0.000	0.000	0.000	0.008	0.000	0.017	0.000	0.000	0.000	0.013	0.000	0.026	0.000
aldi	1.000	0.979	0.970	0.971	0.914	0.971	0.415	0.434	0.302	0.311	0.302	0.311	0.587	0.601	0.460	0.471	0.454	0.471
apple	0.750	1.000	0.000	0.000	0.000	0.000	0.039	0.066	0.000	0.000	0.000	0.000	0.075	0.123	0.000	0.000	0.000	0.000
becks	0.073	0.050	0.081	0.083	0.079	0.057	0.030	0.020	0.030	0.030	0.030	0.020	0.043	0.029	0.044	0.044	0.043	0.030
bmw	0.917	0.917	0.938	0.929	0.933	1.000	0.149	0.149	0.203	0.176	0.189	0.176	0.256	0.256	0.333	0.295	0.315	0.299
carlsberg	0.170	0.176	0.212	0.192	0.160	0.184	0.074	0.083	0.102	0.093	0.074	0.083	0.103	0.113	0.137	0.125	0.101	0.115
chimay	0.077	0.121	0.136	0.121	0.106	0.081	0.036	0.063	0.080	0.071	0.063	0.045	0.049	0.082	0.101	0.090	0.079	0.057
cocacola	0.947	1.000	0.922	0.939	0.946	0.924	0.554	0.577	0.454	0.477	0.538	0.469	0.699	0.732	0.608	0.633	0.686	0.622
corona	0.044	0.024	0.050	0.031	0.050	0.029	0.024	0.012	0.024	0.012	0.024	0.012	0.031	0.016	0.033	0.017	0.033	0.017
dhl	0.860	0.887	0.870	0.944	0.820	0.857	0.350	0.382	0.325	0.276	0.333	0.293	0.497	0.534	0.473	0.428	0.474	0.436
erdinger	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
esso	1.000	1.000	1.000	1.000	1.000	1.000	0.391	0.356	0.402	0.391	0.425	0.368	0.562	0.525	0.574	0.562	0.597	0.538
fedex	0.855	0.980	0.920	0.938	0.959	0.956	0.500	0.511	0.489	0.479	0.500	0.457	0.631	0.671	0.639	0.634	0.657	0.619
ferrari	0.778	0.750	0.697	0.818	0.677	0.864	0.384	0.370	0.315	0.247	0.288	0.260	0.514	0.495	0.434	0.379	0.404	0.400
ford	1.000	1.000	0.929	1.000	0.929	1.000	0.197	0.250	0.171	0.184	0.171	0.197	0.330	0.400	0.289	0.311	0.289	0.330
fosters	0.300	0.372	0.339	0.400	0.339	0.344	0.184	0.163	0.204	0.122	0.204	0.112	0.228	0.227	0.255	0.188	0.255	0.169
google	0.953	0.938	0.935	0.956	0.956	0.957	0.494	0.542	0.518	0.518	0.518	0.530	0.651	0.687	0.667	0.672	0.672	0.682
guiness	0.892	0.895	0.811	0.914	0.795	0.912	0.337	0.347	0.306	0.327	0.316	0.316	0.489	0.500	0.444	0.481	0.453	0.470
heineken	1.000	0.500	0.500	1.000	0.500	1.000	0.010	0.010	0.010	0.010	0.010	0.010	0.019	0.019	0.019	0.019	0.019	0.019
hp	0.923	0.935	0.906	0.900	0.933	0.966	0.214	0.259	0.259	0.241	0.250	0.250	0.348	0.406	0.403	0.380	0.394	0.397
milka	0.675	0.689	0.701	0.756	0.689	0.740	0.431	0.528	0.523	0.487	0.518	0.477	0.526	0.598	0.599	0.593	0.591	0.580
nvidia	0.100	0.167	0.071	0.143	0.167	0.143	0.009	0.009	0.009	0.009	0.018	0.009	0.016	0.017	0.016	0.017	0.032	0.017
paulaner	0.614	0.623	0.561	0.574	0.596	0.596	0.343	0.373	0.314	0.304	0.304	0.304	0.440	0.466	0.403	0.397	0.403	0.403
pepsi	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
rittersport	0.841	0.827	0.825	0.843	0.837	0.851	0.520	0.539	0.510	0.500	0.505	0.505	0.642	0.653	0.630	0.628	0.630	0.634
shell	0.946	0.921	0.850	1.000	0.861	1.000	0.365	0.365	0.354	0.198	0.323	0.229	0.526	0.522	0.500	0.330	0.470	0.373
singha	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
starbucks	1.000	1.000	1.000	1.000	1.000	1.000	0.326	0.305	0.295	0.295	0.305	0.295	0.492	0.468	0.455	0.455	0.468	0.455
stellaartois	0.957	0.905	0.947	0.950	0.905	0.950	0.253	0.218	0.207	0.218	0.218	0.218	0.400	0.352	0.340	0.355	0.352	0.355
texaco	0.625	0.421	0.500	0.588	0.647	0.529	0.114	0.091	0.125	0.114	0.125	0.102	0.192	0.150	0.200	0.190	0.210	0.171
tsingtao	0.833	0.571	0.714	0.667	0.833	0.667	0.046	0.037	0.046	0.037	0.046	0.037	0.087	0.069	0.086	0.070	0.087	0.070
ups	0.959	0.980	0.977	0.976	0.977	0.930	0.522	0.533	0.478	0.456	0.467	0.444	0.676	0.691	0.642	0.621	0.632	0.602
μ	0.628	0.613	0.575	0.614	0.583	0.610	0.228	0.237	0.221	0.206	0.221	0.204	0.316	0.325	0.306	0.293	0.307	0.292
σ	0.394	0.392	0.388	0.407	0.387	0.413	0.193	0.203	0.183	0.179	0.187	0.179	0.251	0.260	0.240	0.238	0.244	0.239

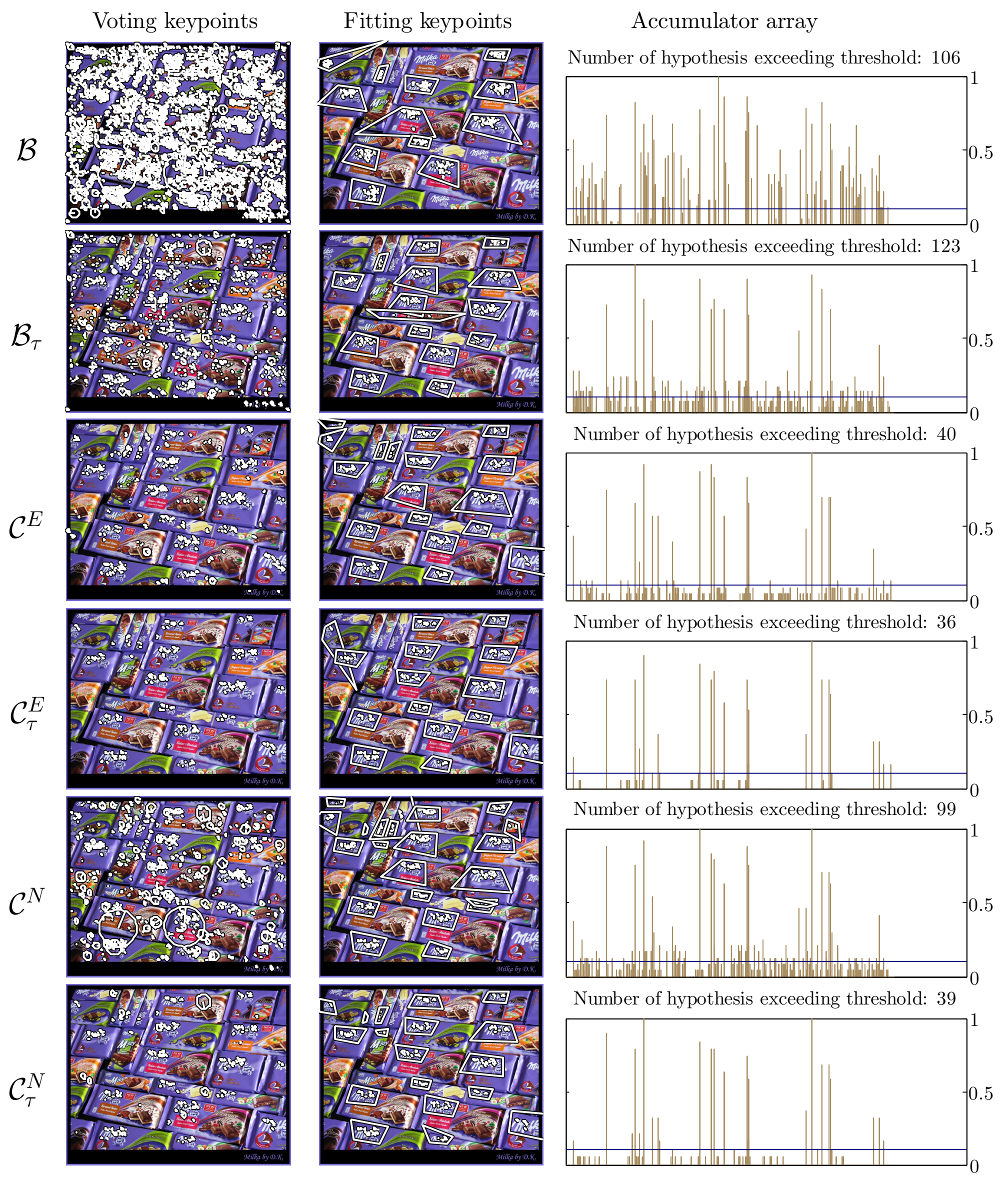


Figure 6.6: Results on image Milka 4733039798.jpg. The potential of the constellation models, in the case of $\mathcal{C}^E$, is clearly visible, since it is able to detect logotypes in the upper left, and bottom right corners where the basic model fails, due to to presence of significant noise in the case of $\mathcal{B}$, and due to lack of information in the case of $\mathcal{B}_\tau$.

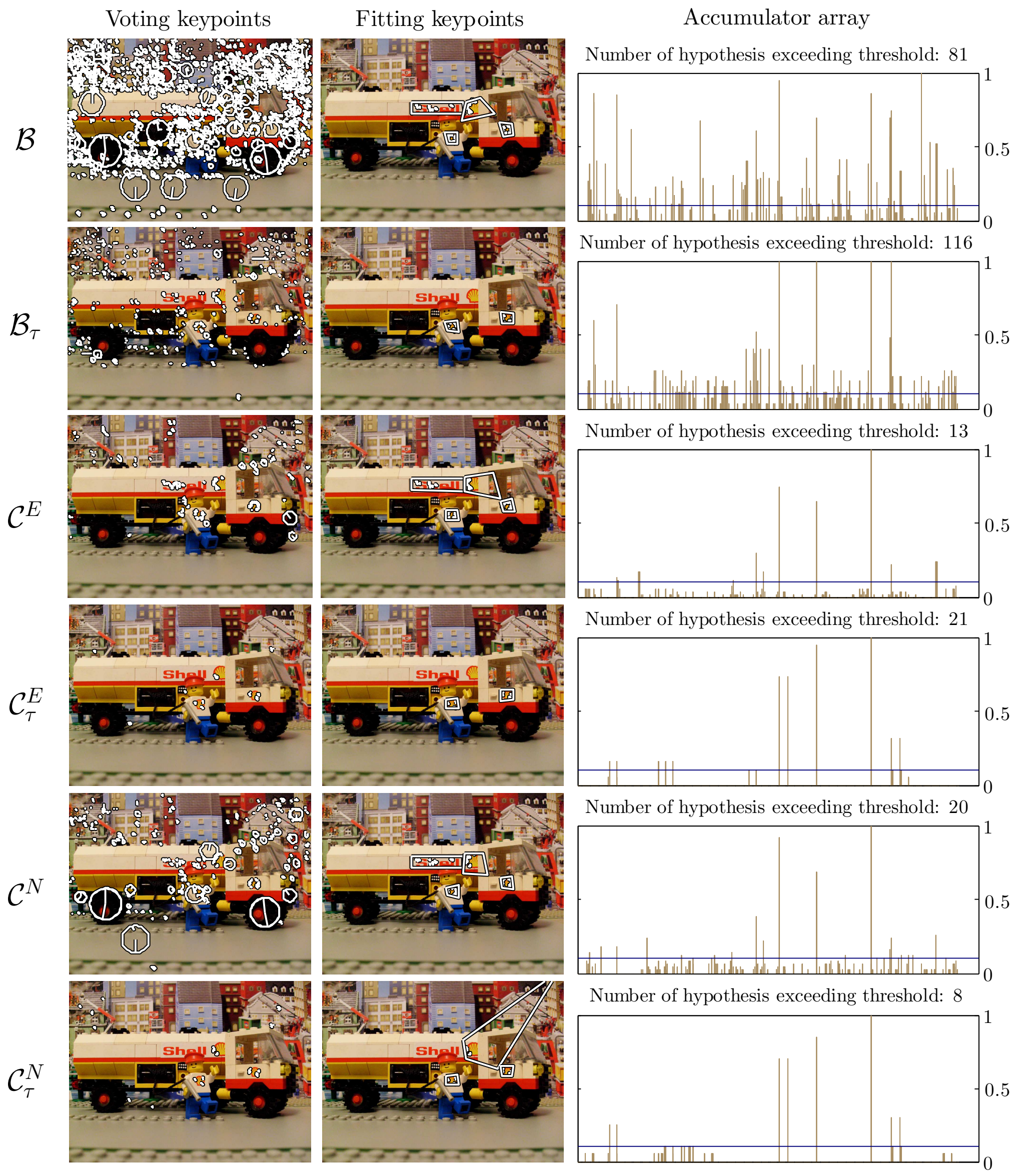


Figure 6.7: Results on image Shell 2178487983.jpg. Constellation models $\mathcal{C}^E$ and $\mathcal{C}^N$, even without the use of the similarity threshold, substantially reduce the amount of noisy keypoints, i.e., accumulator array noise, when compared to basic model $\mathcal{B}$ and even $\mathcal{B}_\tau$, with the use of the similarity threshold. Additionally, only the constellation models are able to detect all logotypes in the test image.

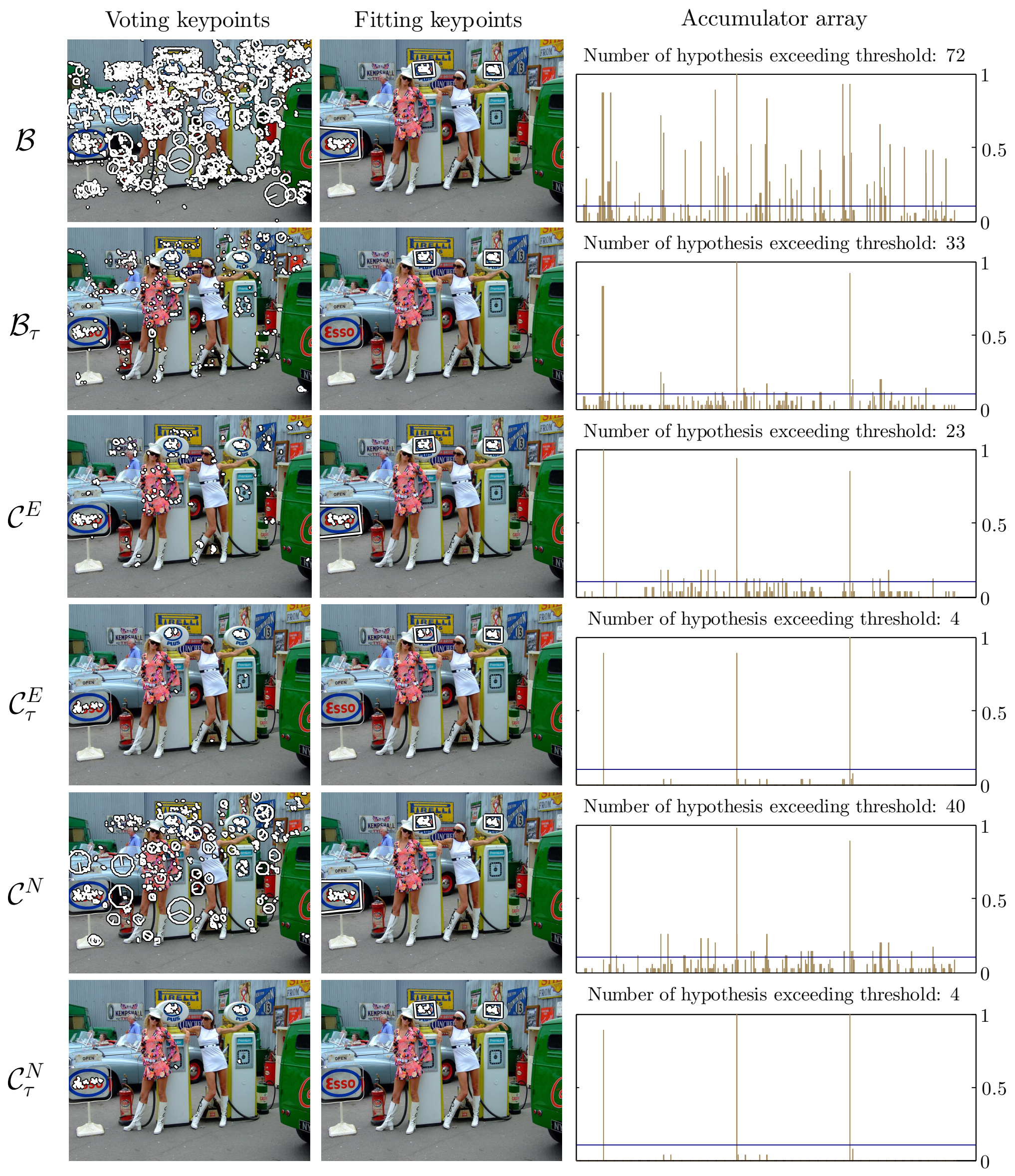


Figure 6.8: Results on image Esso 1305435891.jpg. The filtering capability of constellation models $\mathcal{C}^E$ and $\mathcal{C}^N$, without the use of the similarity threshold, is similar to the performance of the basic model with the use of the similarity threshold $\mathcal{B}_\tau$. All three accumulator arrays have a similar degree of noise, but only the constellation models detect the far left object, additionally to reducing accumulator array noise.

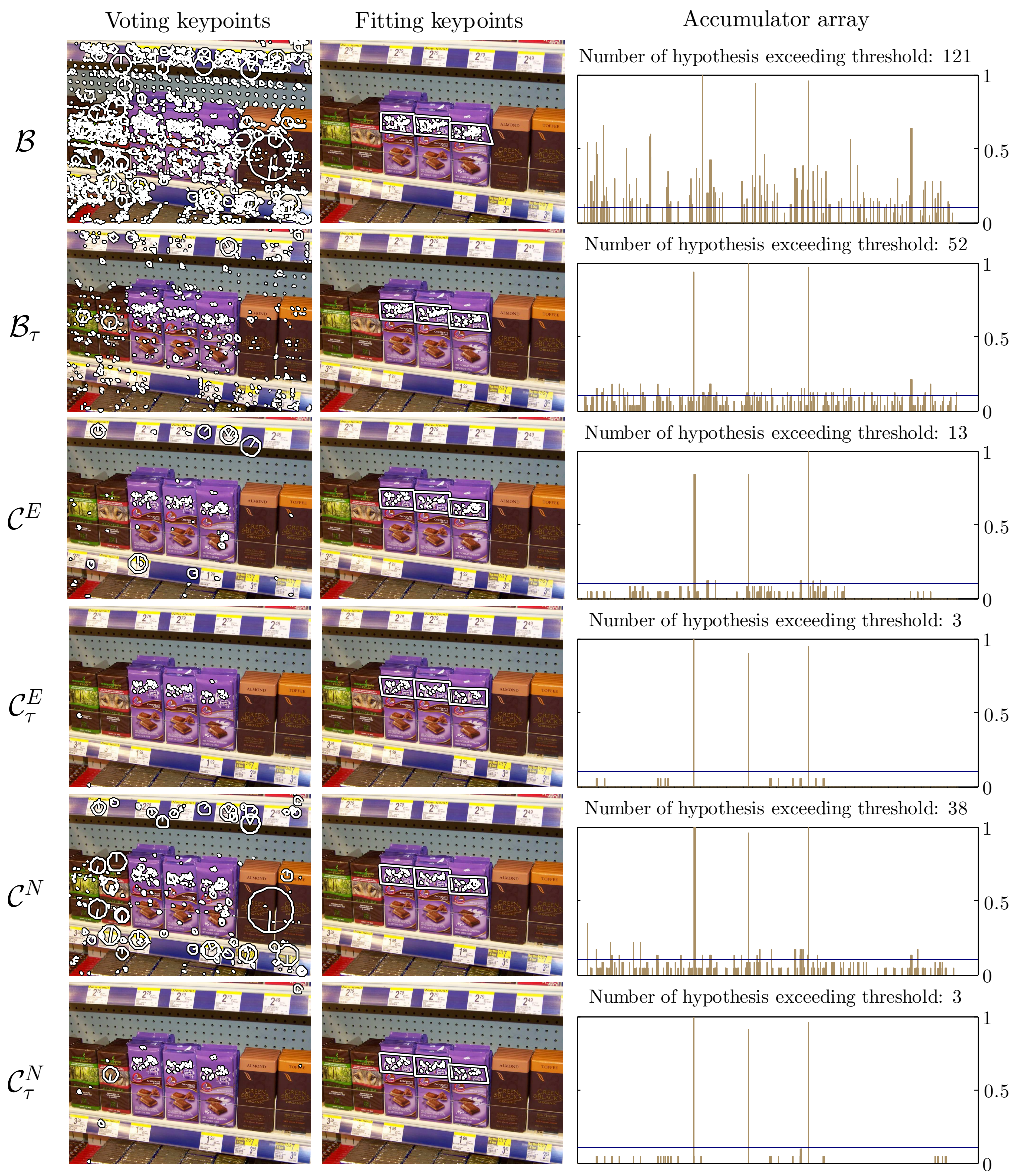


Figure 6.9: Results on image Milka 3244960835.jpg. Constellation models with the use of the similarity threshold $\mathcal{C}^E_\tau$ and $\mathcal{C}^N_\tau$ essentially reduce the number of object location hypothesis in the accumulator array, to the exact number of objects located in the test image. Essentially all noise in the accumulator array gets filtered out, i.e., only keypoints corresponding to the object of interest are left, while all other keypoints get filtered out.

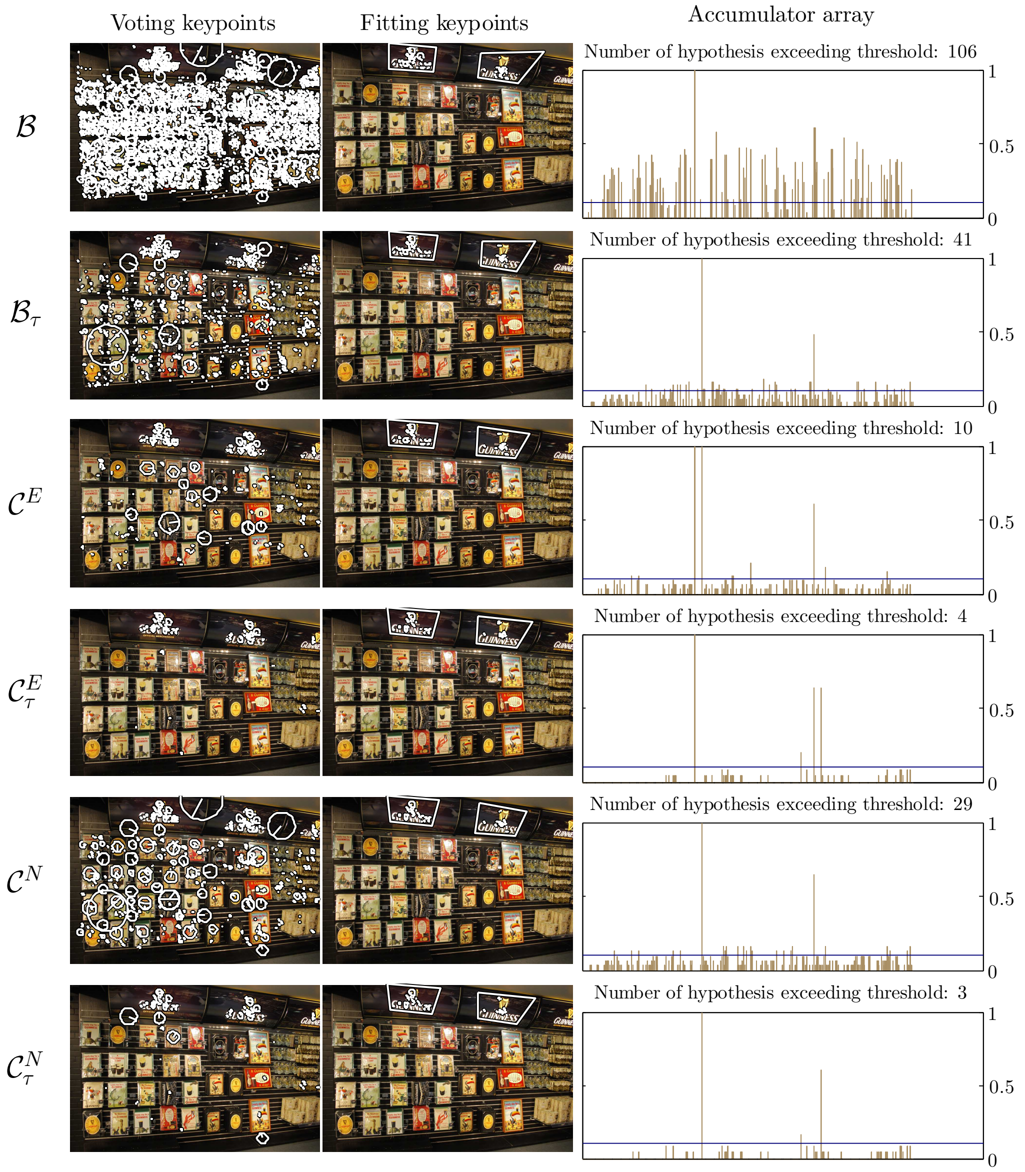


Figure 6.10: Results on image Guiness 4746900318.jpg. The potential of the constellation model is visible in test images with a high degree of background clutter. Additionally to filtering out noisy keypoints, detection accuracy is improved to some extent.

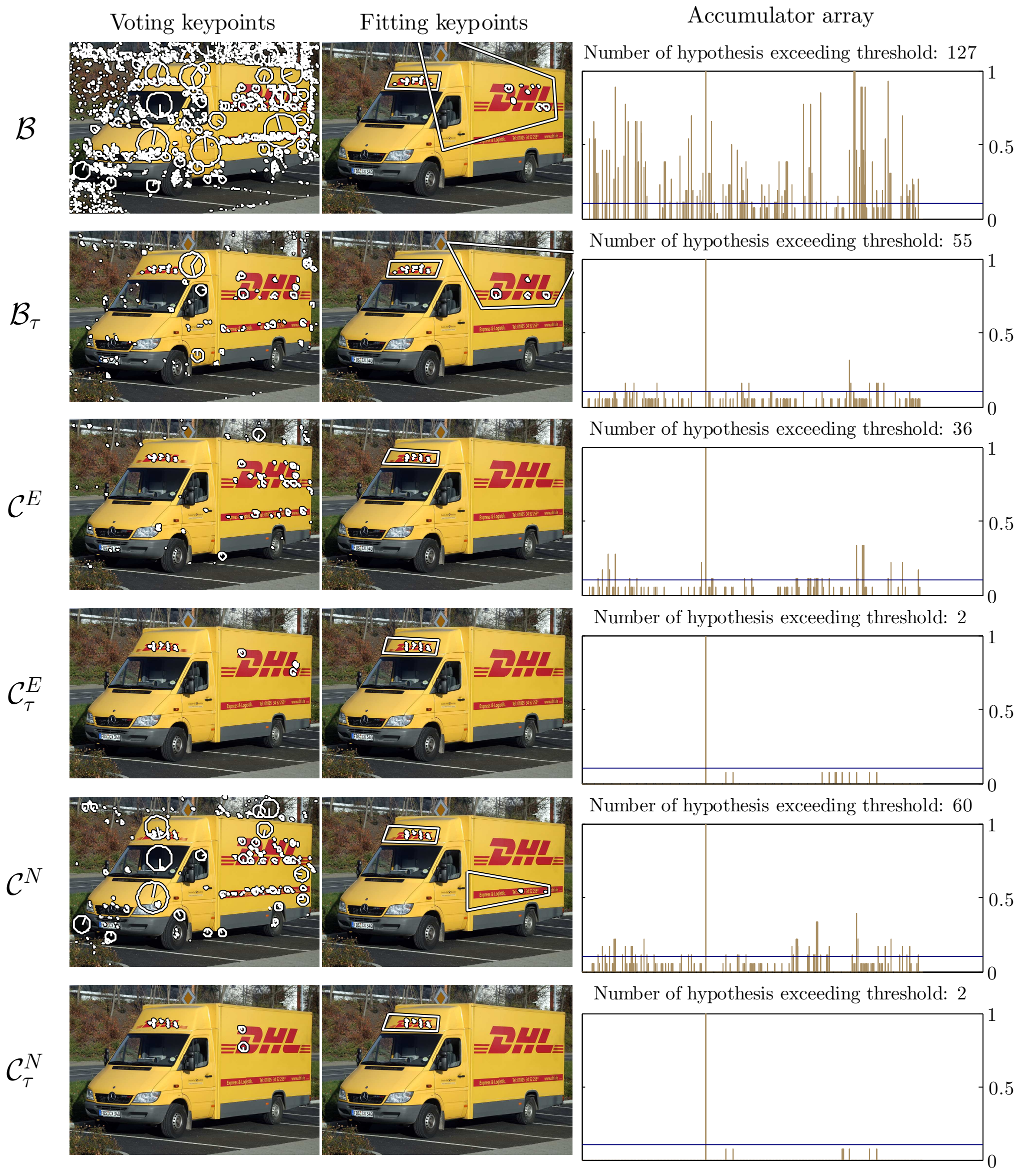


Figure 6.11: Results on image DHL 357147293.jpg. Depiction of constellation model shortcomings when encountering objects, subjected to stronger perspective transformations. Although constellation models address object deformations to some extent, a strongly deformed geometry between detected keypoints will nevertheless conflict with the encoded geometry restrictions in the model, resulting in the incapability to detect objects under stronger transformations.

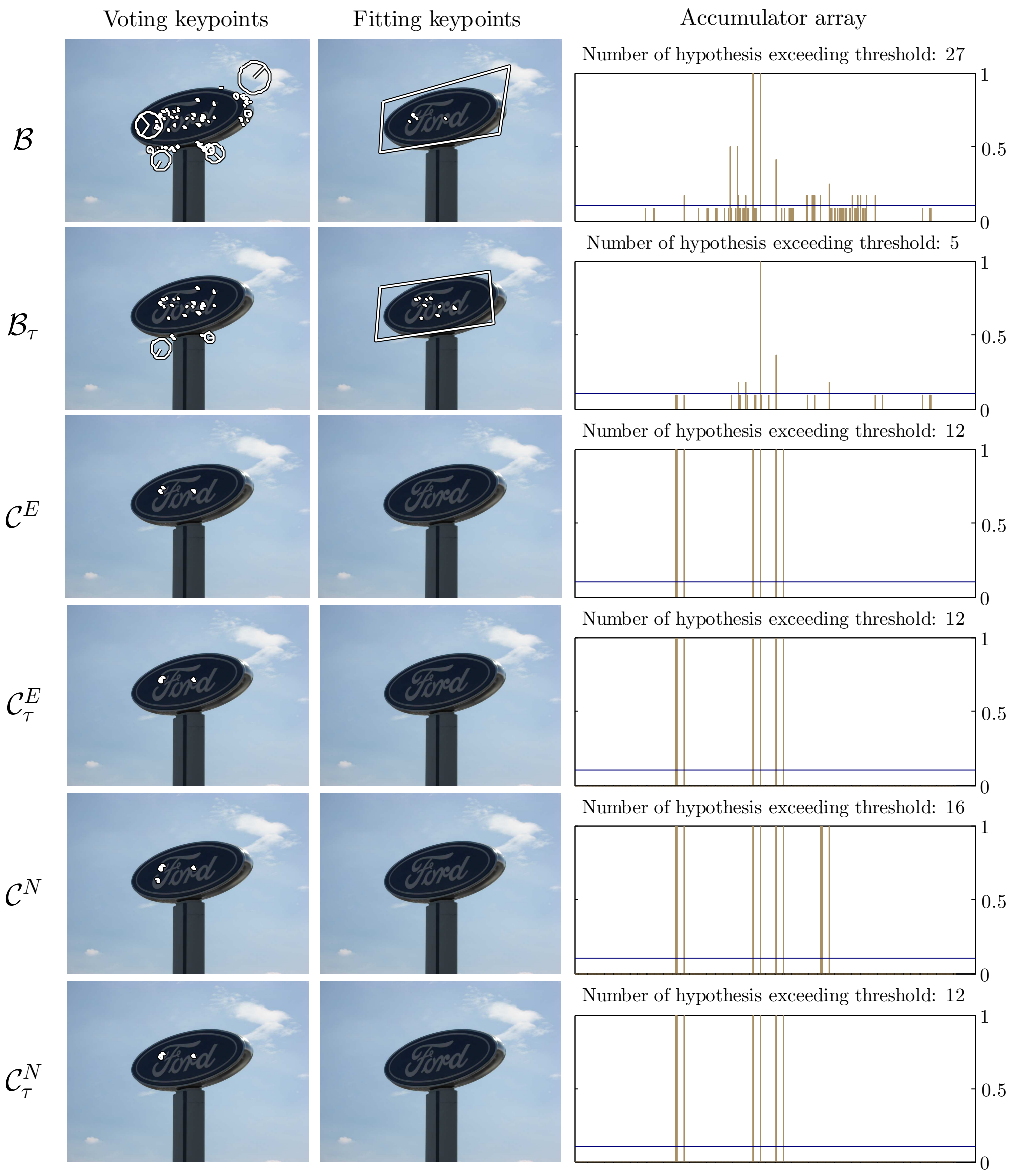


Figure 6.12: Results on image Ford 2390642427.jpg. Depiction of a case where the constellation models prove incapable of detecting the object of interest, since the deformed geometry between detected keypoints conflicts with the encoded geometry restrictions in the constellation model.

Chapter 7

Conclusion

We deal with the problem of one-shot learning and propose a constellation model for the construction of object models as an enhancement of basic feature-based object detection methods. In contrast to the use of the similarity threshold, the proposed constellation model aims in filtering out mismatched features and producing clearer object location hypothesis by the use of pure geometry. In terms of the onte-shot learning restriction a key advantage of the constellation model is that constellations are computed and utilized "on top" of detected features.

Our model exploits the unified representation of most feature extraction methods, enabling it to work independent of the feature-detection algorithm itself. In order to effectively exploit the spatial relations between detected features, these need to be encoded in a matter that allows for their utilization in the object detection process. A constellation should thus span a certain region of an object in order to obtain a constellation with enough features to achieve a distinctive representation of a local region, and at the same time, retain global robustness to potential object occlusions and distortions. Because of this, multiple constellations that model local object-structures are encoded for an arbitrary object, achieving a global coverage. Another aspect is that an encoded constellation cannot merely rigidly model the spatial relations between features, but must allow for a certain degree of deformation within the constellations itself in order to address potential object deformations. For this, a special model that encodes spatial relations between features and also allows for their deformation needs to be constructed. In order to utilize the encoded constellations, a new step needs to be implemented in the detection process which is able to exploit the encoded spatial relations to the benefit of the object detection algorithm. Ideally this leads to a reduction of the number of regions of interest, i.e., object location hypothesis which need verification.

In the context of one-shot learning, the proposed constellation model requires a single planar training image for model construction as it implicitly models the expected deformations of the object. Constellation model construction proceeds as follows. Initially, features are extracted from a test image depicting the object of interest. For each feature, a constellation is obtained by encoding the spatial relations between the neighbouring features. For each constellation, the expected variations of features are modelled in order to obtain deformable constellations, which are able to address local region deformations. The expected feature variation are modelled by two different approaches, i.e., empirically and numerically. All constellations are encoded in a common space, i.e., the unit space. Similar variations of features in the unit space are merged together in order to reduce the total number of encoded variations. Additionally, the center point of the object, the object bounding-box and an intensity template of the object are stored in the model.

Two variations of the proposed constellation model, with empirically and numerically modelled variance, and the basic feature model, all with and without the similarity threshold were evaluated on the challenging real-world dataset FlickrLogos-32 [28], depicting logotypes in real-world environments. Overall, the proposed constellation models reduce the number of mismatched features, without significantly affecting detection performance. The best variation of the constellation model is the constellation model with empirically determined feature variance. By reducing the number of mismatched features, the constellation model reduces the number of potential object location hypothesis which need to be verified, achieving a substantial noise reduction compared to the similarity threshold, even in highly cluttered environments. The solely on geometry based constellations are computed and utilized "on top" of detected features, so any feature extraction method whose keypoints are represented by a center location, scale and orientation can be enhanced with the proposed constellation model.

Although multiple different objects can be encoded in the same constellation model, in practice this would not prove useful since at a certain point constellations would lose their discriminativeness, starting to filter out less and less noisy keypoints for a given object, as they would start accounting for a group of otherwise different objects. A better practice would be to firstly determine most probable objects located on a test image, for example with a bag-of-words method, and later apply constellation models for specific objects, most likely located on the test image.

7.1 Future work

The proposed constellation model produces a significant amount of feature variations, which need to be verified in the constellation filtering process. This represents a bottleneck of the constellation model, since for each detected feature, matched to a feature in the constellation model, a number of distributions need to be verified. The process of verifying whether a feature is within a given distribution should be optimized in future work, since the process is highly paralyzable.

The incapability of the constellation model to address stronger object deformations, due to the encoded geometry restrictions in the model itself, is also an opened problem which would need to be addressed in future work. One solution could be to predetermine the relative position of an object in a test image and accordingly "deform" the constellation model to coarsely fit on the relative position of the object. Another solution would be to use multiple training images depicting stronger object deformations, since the constellation model is easily scaled to an arbitrary number of objects. An analogy to the first solution would be to use the estimated transformations between the training images and generate constellation deformations accordingly.

A possible application of the proposed constellation model could be in querying large image collections. In a given image, the constellation model would be constructed on a marked region of interest and utilized to find matching instances in a large collection of images.

Bibliography

[1] C. Harris, M. Stephens, A combined corner and edge detector, in: Proceedings of Fourth Alvey Vision Conference, 1988, pp. 147–151.

[2] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, 2005, pp. 886–893.

[3] L. Fei-fei, R. Fergus, P. Perona, One-shot learning of object categories, in: IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 28, 2006.

[4] O. Chum, J. Philbin, J. Sivic, M. Isard, A. Zisserman, Total recall: Automatic query expansion with a generative feature model for object retrieval, in: IEEE 11th International Conference on Computer Vision (ICCV), 2007, pp. 1–8.

[5] Y. Zhang, Z. Jia, T. Chen, Image retrieval with geometry-preserving visual phrases, in: IEEE Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR), 2011, pp. 809–816.

[6] N. Buch, S. Velastin, J. Orwell, A review of computer vision techniques for the analysis of urban traffic, in: IEEE Transactions on Intelligent Transportation Systems, Vol. 12, 2011, pp. 920–939.

[7] P. Piccinini, A. Prati, R. Cucchiara, Real-time object detection and localization with sift-based clustering, in: Image and Vision Computing, Vol. 30, 2012, pp. 573–587.

[8] M. Merler, C. Galleguillos, S. Belongie, Recognizing groceries in situ using in vitro training data, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007, pp. 1–8.

[9] E. Hsiao, A. Collet, M. Hebert, Making specific features less discriminative to improve point-based 3d object recognition, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 2653–2660.

[10] T. Winlock, E. Christiansen, S. Belongie, Toward real-time grocery detection for the visually impaired, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2010, pp. 49–56.

[11] D. G. Lowe, Object recognition from local scale-invariant features, in: IEEE Computer Society Proceedings of International Conference on Computer Vision (ICCV), Vol. 2, 1999, pp. 1150–1157.

[12] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: IEEE Computer Society Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, 2001, pp. 511–518.

[13] R. Brunelli, Template Matching Techniques in Computer Vision: Theory and Practice, Wiley Publishing, 2009.

[14] J. Khan, S. Bhuiyan, R. Adhami, Image segmentation and shape analysis for road-sign detection, in: IEEE Transactions on Intelligent Transportation Systems, Vol. 12, 2011, pp. 83–96.

[15] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints, in: Workshop on Statistical Learning in Computer Vision (ECCV), 2004, pp. 1–22.

[16] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-up robust features (surf), in: Computer Vision and Image Understanding (CVIU), Vol. 110, 2008, pp. 346–359.

[17] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part based models, in: IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 32, 2010, pp. 1627–1645.

[18] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, in: IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 27, 2005, pp. 1615–1630.

[19] S. Lazebnik, C. Schmid, J. Ponce, Semi-local affine parts for object recognition, in: British Machine Vision Conference (BMVC), Vol. 2, 2004, pp. 779–788.

[20] S. Kim, K. jin Yoon, I. S. Kweon, Object recognition using generalized robust invariant feature and gestalt law of proximity and similarity, in: IEEE Workshop on Perceptual Organization in Computer Vision and Pattern Recognition (CVPR), 2006.

[21] Y. Ke, R. Sukthankar, Pca-sift: A more distinctive representation for local image descriptors, in: IEEE Computer Society Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR), 2004, pp. 506–513.

[22] M. Toews, W. Wells, Sift-rank: Ordinal description for invariant feature correspondence, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 172–177.

[23] S. Romberg, R. Lienhart, Bundle min-hashing for logo recognition, in: Proceedings of the 3rd ACM International Conference on Multimedia Retrieval (ICMR), 2013, pp. 113–120.

[24] S. Romberg, R. Lienhart, Bundle min-hashing, in: International Journal of Multimedia Information Retrieval, Vol. 2, 2013, pp. 243–259.

[25] N. Friedman, D. Geiger, M. Goldszmidt, G. Provan, P. Langley, P. Smyth, Bayesian network classifiers, in: Machine Learning, 1997, pp. 131–163.

[26] C. Cortes, V. Vapnik, Support-vector networks, in: Machine Learning, Vol. 20, 1995, pp. 273–297.

[27] M. A. Fischler, R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, in: Communications of the ACM, Vol. 24, 1981, pp. 381–395.

[28] S. Romberg, L. G. Pueyo, R. Lienhart, R. van Zwol, Scalable logo recognition in real-world images, in: Proceedings of the 1st ACM International Conference on Multimedia Retrieval (ICMR), 2011, pp. 25:1–25:8.

[29] Y. Yang, I. Saleemi, M. Shah, Discovering motion primitives for unsupervised grouping and one-shot learning of human actions, gestures, and expressions, in: IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 35, 2013, pp. 1635–1648.

[30] C. H. Lampert, H. Nickisch, S. Harmeling, Attribute-based classification for zero-shot visual object categorization, in: IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), Vol. 36, 2014, pp. 453–465.

[31] D. A. Forsyth, J. Ponce, Computer Vision: A Modern Approach, 2nd edition, Prentice Hall Professional Technical Reference, 2011.

[32] D. H. Ballard, Generalizing the hough transform to detect arbitrary shapes, in: Readings in Computer Vision: Issues, Problems, Principles, and Paradigms, 1987, pp. 714–725.

[33] R. O. Duda, P. E. Hart, Use of the hough transformation to detect lines and curves in pictures, in: Communications of the ACM, Vol. 15, 1972, pp. 11–15.

[34] P. H. S. Torr, A. Zisserman, Mlesac: A new robust estimator with application to estimating image geometry, in: Computer Vision and Image Understanding (CVIU), Vol. 78, 2000, pp. 138–156.

[35] M. Everingham, L. Gool, C. K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, in: International Journal of Computer Vision (IJCV), Vol. 88, 2010, pp. 303–338.