Karlsruher Institut für Technologie

Masterarbeit

Verschnittoptimierung für zweidimensionale Prototypenfertigungsprozesse

Jonas Fuchs B.Sc.

21.06.2015 bis 18.12.2015

Betreuung: Herr Prof. Dr. rer. nat. Peter Sanders

Institut für Theoretische Informatik, Algorithmik II
Fakultät für Informatik
Karlsruher Institut für Technologie

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Der technische Fortschritt in den letzten Jahrzehnten ist im Bereich der Fertigungsverfahren bei weitem nicht im Rahmen der großindustriellen Anwendung geblieben, sondern ermöglicht es heute selbst Hobbymodellbauern hochwertige Teile auf Kleinstmaschinen fertigen zu lassen. Durch den verkleinerten Nachbau von Großmaschinen sind heute Fertigungsverfahren wie CNC-Fräsen, additive Verfahren wie 3D-Druck und Materialschnitt per Laserstrahl mit einem Budget unter 10.000 € als fertige Maschine zu erwerben. Viele Maschinenelemente orientieren sich dabei an den Lösungen der Großserie und profitieren so vom weit fortgeschrittenen Entwicklungsstand dieser Maschinen. Doch nicht bei allen Elementen der Geräte lässt sich dieser Schritt durchführen. Der Einsatzbereich ist derart anders gestaltet, dass z.B. die Automatisierung von Optimierungsprozessen, wie der platzeffizienten Anordnung der gewünschten Schnittkonturen, nicht direkt übertragen werden kann.

Im professionellen Arbeitsalltag finden mittlerweile auch die kleineren Geräte für 2D-Zuschnitt Einzug und werden in der Kleinserienfertigung und dem Prototypenbau eingesetzt. Zusammenfassend wird von Rapid-Prototying gesprochen, da man durch die Fertigung und Selbstausführung vor Ort nur eine sehr kurze Fertigungsdauer vorfindet.

Als Anwender derartiger Technologien ist man meist nicht in automatisierte Prozesse eingebunden und die Fertigungsmaschine hat eine relativ geringe Auslastung und wechselnde Nutzer. Ein damit aufkommendes Problem ist in vielen Fällen die Verwaltung, Ordnung und Nutzung von Roh- und Restmaterial. Einhergehend damit ist die Platzierung von Fertigungsteilen auf zur Verfügung stehendem, bereits angeschnittenem Restmaterial. Im Rahmen dieser Arbeit sollen für das Umfeld der Prototypen- und Kleinserienfertigung passende Algorithmen gefunden werden um die Materialverwendung effizienter zu gestalten.

1.2 Anwendungsbereiche

Um das Umfeld der Arbeit besser zu verstehen, werden im Folgenden Anwendungsbereiche und zugrundeliegende Technologien eingeführt und erläutert.

1.2.1 Trennverfahren Im Rahmen von Fertigungsprozessen wird in den meisten Fällen Rohmaterial verarbeitet und veredelt. In vielen Branchen wird das Halbzeug dabei in flacher Form angeliefert. Im Regelfall als Platten, Rollen oder natürlich begrenzte Formstücke wie Leder. Zur Weiterverarbeitung wird das Rohmaterial dabei gezielt in kleinere Stücke getrennt, es wird also innerhalb einer 2D-Ebene getrennt. Dabei gibt es sehr viele verschiedene Materialien und Verfahren, welche zum Einsatz kommen.

Blech- / Stahlzuschnitt: Ein klassisches Anwendungsgebiet für Flachmaterial ist die Weiterverarbeitung von Stahlblechen. Gerade durch die intensive Verwendung des Werkstoffs in der Automobil-, Luftfahrt- und Schiffsbaubranche sind viele Verfahren geschaffen worden, welche für die Serienprozesse aufwändige händische Trennarbeiten ersetzen. Hierbei werden Trennverfahren wie Stanzen, Laserschneiden, Wasserstrahlschnitt und Scheren verwendet. Angeliefert wird das Material je nach Branche als Plattenmaterial mit bis zu 2×6 Metern.

Textilindustrie: Früher als handwerkliche Arbeit mit der Handschere durchgeführt, wurde für die großindustrielle Fertigung von Kleidung schnell auf Automatisierung und Hilfsmaschinen gesetzt. Angeliefert wird der Rohstoff Textil als breites Tuchmaterial auf einer Rolle und ist somit ein sehr langer Streifen Stoff. In diesem Umfeld hat das klassische Strip-Packaging-Problem seinen Ursprung, bei dem die gewünschten Textilzuschnitte in großer Stückzahl auf einen Streifen mit fester Breite aber quasi beliebiger Länge platziert werden -auf sogenannten Endlosbahnen.

Leder: Als Sonderfall der Bekleidungsbranche haben Gerber und Täschner ein komplexes Problem des zweidimensionalen Zuschnitts. Durch die natürliche Beschaffenheit von Leder sowohl in Form, als auch in Oberflächenqualität ist viel Sorgfalt für Platzierung und Zuschnitt gefordert. Durch den hohen Preis des Rohstoffs Leder haben hier computergestützte Optimierungsansätze früh Einzug gehalten.

Abbildung 1. Laseranlage der Firma Trumpf für industrielle Fertigung[Trum15]

Folien- / Papierzuschnitt: Vor allem in der Werbebranche mit einem hohem Grad an Individualisierung ist der nicht-triviale Zuschnitt von Aufklebern, Aufstellern und Ähnlichem wichtiger Bestandteil, der durch zweidimensionale Trennverfahren umgesetzt wird. Hierbei kommen vor allem Schnittverfahren über (Schlepp-)Messer und Scheren zum Einsatz.

Gravur und Markierung: Auch im Bereich der Werbe- und Kreativbranche sind zweidimensionale Oberflächenbearbeitungen zur Aufbringung von Bildern oder Beschriftungen mit Lasern im Trend. Da es sich meist um die Veredlung von bereits bestehenden Rohprodukten handelt, bei denen an eine bereits definierte Position ein Motiv aufgebracht wird, ist eine automatisierte Platzierung unüblich. In diesem Anwendungsfall wird das Material aber nur oberflächlich angeschnitten, durch den fehlenden Durchbruch handelt es sich also nicht um ein tatsächliches Trennverfahren hat aber einige Verwandtschaften.

Weitere: Durch die zunehmende Materialvielfalt verschwimmen die Anwendungsbereiche der Trenntechnologien zunehmen, sodass z.B. Schnittverfahren auch für technische Gewebe wie Kohlefasermatten verwendet werden, meist kombiniert mit anderen Schneidemitteln wie Rollmessern, Scheren oder Hubmessern.

Verwandte Branchen: Eine der wichtigen Teilaufgaben bei der Verkleinerung von Rohmaterial ist die Platzierung der Teilstücke. Diese räumliche Anordnung ist von der Aufgabe und Problemstellung sehr verwandt mit Platzierungsproblemen aus der Logistikbranche. Das Packen von Seecontainern oder Lieferfahrzeugen mit Gütern wird mit ähnlichen Problemlösungsvorgehen bewältigt. Die Geometrien der zu platzierenden Objekte sind meist einfacher, da rechtwinklig begrenzt, durch die dreidimensionale Komponente ist die Komplexität des Problems dennoch enorm.

Allgemein: Wie in den vorangegangenen Anwendungsfällen von 2D-Trennverfahren ersichtlich, handelt es sich meist um Hochstückzahl- oder Hochpreisanwendungen, bei denen Handarbeit aufgrund wirtschaftlicher und qualitativer Vorteile ersetzt wird. Aufgrund dieser Kriterien finden wir in den industriellen Umfeldern teure Fertigungsmaschinen mit hoher Auslastung und meist fest zugeordneten und geschulten Mitarbeitern, welche diese bedienen. Deren Tätigkeit besteht im Bestücken der Maschine mit Rohmaterial und Fertigungsdaten und der Entnahme der zugeschnittenen Teile. Um die Wirtschaftlichkeit zu erhöhen, wird versucht diese Prozesse möglichst kurz zu gestalten und stark zu automatisieren. Der restliche Kostenfaktor ist der Anteil an Verschnittfläche im Verhältnis zur Fläche der herausgeschnittenen Teile.

1.2.2 Laserschnittverfahren Im Rahmen dieser Arbeit wird stellvertretend für 2D-Fertigungsverfahren der Schnitt per Laserstrahl näher betrachtet. Hierbei lassen sich üblicherweise im niedrigen Leistungsbereich vor allem Kunststoffe und Hölzer schneiden. Im höheren Leistungsbereich können auch Metalle geschnitten werden. Der Laserschnitt findet dabei berührungslos und nahezu kraftfrei[1] statt. Im Gegensatz zu z.B. Stanzverfahren wird kein spezielles Werkzeug benötigt, sodass auch kleine Stückzahlen kostengünstig gefertigt werden können. Von wirtschaftlicher Seite betrachtet, spielt die Dauer des Schnittes eine große Rolle. Abrechnungskennzahl ist daher meist die aufsummierte Schnittlänge aller Teile, da die Schnittgeschwindigkeit pro Material gleichbleibend ist. Verschleißbehaftet ist

[1] Durch Zublasung von Luft und Verspannungen durch Hitzeeinwirkung können kleine Kräfte im Material auftreten.

vor allem die Laserröhre und die Mechanik der Maschine und stellt daher einen großen laufenden Kostenpunkt dar. Im Rahmen der Anwendung fällt neben der Platzierung der Teile meist noch eine Optimierung der Schnittreihenfolge (vgl. Abbildung 2) an, mit der die Schnitttoleranzen verbessert werden. Wenn Reihenfolge und Position feststehen, wird zudem eine Pfadoptimierung durchgeführt um unnötige Verfahrwege zu vermeiden.

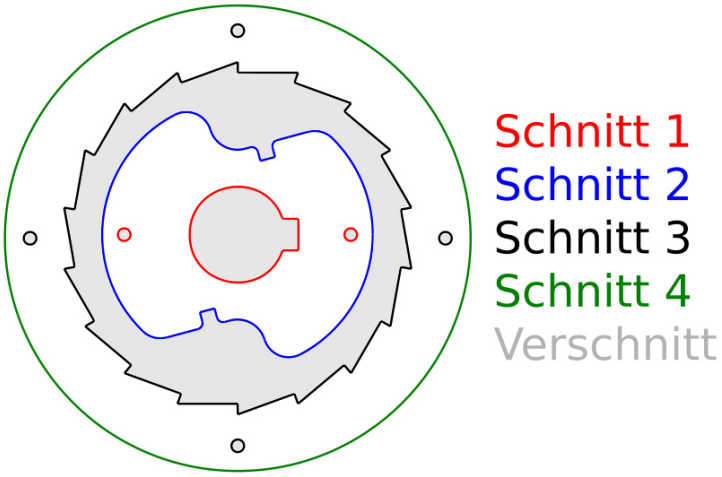

Abbildung 2. Zu schneidende Konturen von ineinander verschachtelten Teilen

Entwicklungsfortschritt und Maschinenverfügbarkeit: In den letzten Jahren wurden die Stückzahlen und Technologiefortschritte im Umfeld der Laserfertigungsmaschinen stetig größer. Dies ermöglichte es, kompaktere und spezialisierte Maschinen zu konstruieren und neue Geschäftsfelder zu erschließen. Heute schon sind in der Preisspanne 1.000 € bis 100.000 € verschiedenste Modelle erhältlich. Viele davon sind angepasst an die Verwendung in Büroumgebungen und durch Laien bedienbar. Kleinste Werkzeugbettgrößen sind damit bei etwa $30 \times 40cm$. Statt der mehrere Megawatt leistungsstarken Maschinen haben diese nur Leistungsbereiche von 10-200 Watt und damit einen beschränkten Einsatzbereich bezüglich verwendbarer Materialien. Aufgrund des Anwendungsgebiets und des kurzen Fer-

tigungszeitraums spricht man in diesem Bereich von sogenanntem „Rapid Prototyping“.

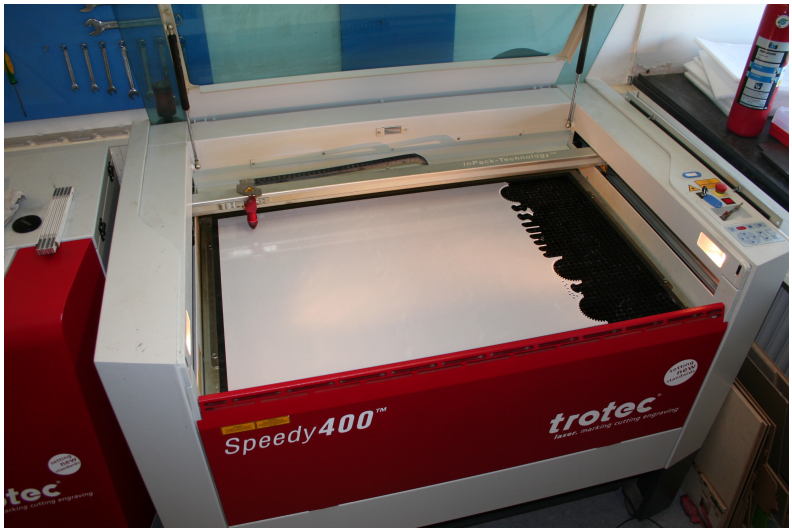

Abbildung 3. Lasersystem der Firma Trotec für Kleinserien und Prototypenbau (oberes Preisspektrum)

Verwendung in der Entwicklung von Prototypen: Durch die einfachere Bedienbarkeit und kostengünstigere Beschaffung gegenüber Großindustriemaschinen haben kleine Lasermaschinen Einzug in andere Branchen erhalten. Speziell im Modell- und Prototypenbau der Design-, Architektur- und Maschinenbaubranche sind die Maschinen willkommene Hilfsmittel, um Funktions- oder Gestaltprototypen zu erstellen. Hierbei stehen die kurzfristige Verfügbarkeit und kurze Iterationszyklen im Vordergrund. Meist werden die Geräte direkt für eine Entwicklungsabteilung angeschafft oder an zentraler Stelle in einem Unternehmen bereitgestellt. Im Vergleich zu den großindustriellen Anwendungen ist die Maschinenauslastung wesentlich niedriger und die Größe von Fertigungsaufträgen geringer. Faktoren wie Schnittdauer und Einrichtungszeit stehen daher deutlich im Hintergrund. Auch sind die Systeme weniger in automatisierte Gesamtsysteme eingefasst, sodass wesentlich mehr manuelle Arbeit erledigt werden muss. Anstatt großer Fertigungsaufträge werden meist Kleinstauf-

Abbildung 4. Importiertes Lasersystem für Hobby und Kleingewerbe[Hobl15]

träge über den Tag verteilt ausgeführt. Aber auch im privaten Bereich sind die günstigen Einsteigermodelle für Bastler ein hochgeschätztes Werkzeug.

1.3 Verwandte Arbeiten

Die Anordnung von Schnittkonturen auf Rohmaterial ist in der Industrie ein gängiges Problem, das im Zuge von Industrialisierung, Globalisierung und damit einhergehenden Hochstückzahlprodukten in vielen Branchen auftritt. Aufgrund des enormen finanziellen Einsparpotentials durch Zeitersparnis und höhere Materialausnutzung, wurden schon früh (vgl. [Art 66]) theoretische Überlegungen zu Optimierungen durchgeführt. Die Randbedingungen beinhalteten dabei meist nur rechteckige Konturen und orthogonale Schnitte. Das Platzieren von Rechtecken auf einem Endlosband für das sogenannte Guillotine-Cutting-Verfahren war dabei eine der ersten Anwendungen für den Textilbereich. Bei diesem Verfahren können nur Schnitte getätigt werden, die das Material von Rand zu Rand in einer geraden Linie teilen.

Seither werden Platzierungsverfahren stets verbessert, auf neue Branchen adaptiert und von Randbedingungen befreit, um mehr Flexibilität und Stabilität zu bieten. Burke[BHKW07] zeigt in seiner Arbeit einen umfangreichen Vergleich seines Vorberechnungsansatzes und die Ergebnisse aus verschiedenen Problemstellungen. Einen Laufzeit- und Ergebnisvergleich führen Chen[ChHu07] und Shalaby[ShKa13] durch. Die direkten Ergebnisse müssen jedoch aufgrund der verschiedenen Veröffentlichungsjahre stets unter Berücksichtigung der bei der Durchführung zur Verfügung stehenden Hardware betrachtet werden und behandeln meist nicht die gleichen Testdatensätze.

Neben der Anwendung im Bereich der Fertigung, wird man in der Logistik bei sehr verwandten Platzierungsproblemen, dem sogenannten Packaging, fündig. Im Anwendungsfall geht es meist um das raumoptimale Füllen von Transportcontainern. Auch sehr aktuelle Arbeiten, wie z.B. die von Bansal[BaKh14], beschäftigen sich mit 2D-Packaging von Rechtecken. Speziell für eine sehr große prozentuale Ausnutzung des zur verfügbaren Raumes sind neue Ansätze und Verfahren notwendig.

Auch aus den handwerklichen Anwendungsfällen, bei denen wenig Automation angewendet wird, finden sich wissenschaftliche Arbeiten, die Platzierungsprozesse untersuchen. So hat sich Heistermann[HeLe95] mit den Platzierungsproblemen in der Lederindustrie beschäftigt und grundlegende Prozesse und Ansätze analysiert und erarbeitet. Besonders hervorzuheben sind die Randbedingungen der irregulär begrenzten Teilstücke und Rohmaterialstücke, die für einen Platzierungsalgorithmus eine besondere Komplexität mit sich ziehen. Da dieses Problem auch in anderen Industrien auftritt, setzen sich auch spätere Arbeiten, wie beispielsweise die von Liu[LiHe06] und Pinheiro[PiAJS15], mit der Platzierung irregulär begrenzter Teile auseinander und lösen diese über genetische Algorithmenansätze.

Zur Problemlösung wurden auch schon evolutionäre Algorithmen angewandt. Shalaby [ShKa13] nutzte den Particle-Swarm-Ansatz, um 2D-Platzierungen von irregulär begrenzten Teilen auf langen Materialbahnen durchzuführen und erzielte damit sehr hohe Nutzungsgrade.

Wie an der Vielzahl verwandter wissenschaftlicher Arbeiten zu erkennen ist, sind Platzierungsprobleme schon länger Forschungsbestandteil vieler Algorithmenentwickler. Durch den technischen Fortschritt von Fertigungsverfahren und den berechnenden Maschi-

nen wurden alte Arbeiten überholt und neue Ansätze entwickelt. Um für den Einsatz im Umfeld Prototypenentwicklung und Kleinserienfertigung einen passenden Algorithmus zu entwickeln, bieten diese bekannten Algorithmen einen guten Ausgangspunkt.

1.4 Aufgabenstellung

Für die Verwendung von Laserschneidverfahren im Prototypenbau eines Entwicklungsprozesses soll ein Algorithmus konzipiert und implementiert werden, welcher langfristig in eine Prozesskette integriert werden kann und die Nutzer des Lasersystems unterstützt. Hierzu sollen besonders in der Fertigungsdatenaufbereitung und intelligenten Platzierung bestehende händische Lösungen automatisiert und verfügbare Technologien aus der Großserie auf die Anforderungen eines Rapid-Prototyping-Systems adaptiert werden. Die Ziele liegen dabei auf einer Effizienzsteigerung (Materialausnutzung, Rüstzeit) und kurzen Laufzeiten auf regulären Rechnerarchitekturen.

1.5 Gliederung der Arbeit

Der Einleitung in diesem Kapitel folgt die Vermittlung von Grundlagen aus dem Anwendungsbereich und der Algorithmen zum weiteren Verständnis der Arbeit in Kapitel 2. Die Entwicklung des Algorithmus wird in seinen Aspekten in den darauffolgenden Kapiteln jeweils beleuchtet und stückweise verfeinert. Im 3. Kapitel, der Analyse, werden die Anwendungsbereiche näher erläutert, untersucht und bisherige Lösungen aufgezeigt, um anschließend den Entwurfsprozess und Konzeptentscheidungen in Kapitel 4 zu erläutern. Die Implementierung wird in Kapitel 5 erklärt und deren Leistungsfähigkeit im Validierungkapitel 6 geprüft. Die Arbeit wird dann mit einer Diskussion unter Kaptitel 7. abgeschlossen.

2 Grundlagen

2.1 Problemstellung

Um die Aufgabenstellung exakt bearbeiten zu können, werden nun Randbedingungen und Anforderungen formal definiert. Dies soll eine klare Ordnung für spätere theoretische Überlegungen liefern und die Validierung erleichtern. Die benutzten Schnittoperationen sind geometrisch zu deuten.

Der allgemeine Anwendungsfall beinhaltet eine gegebene Menge an zweidimensionalen Konturen, die als Rohmaterial bezeichnet werden. Zur Vereinfachung wird im Weiteren davon ausgegangen, dass nur eine Kontur C als Rohmaterial R gegeben ist.

$$R = C | C \in \text{2D Kontur}; |R| = 1 \tag{1}$$

Auf diesem Rohmaterial soll eine Menge an Teilen T platziert werden, welche selbst ebenfalls zweidimensionale Konturen sind. Diese Menge muss nicht sortiert sein und kann Elemente mehrfach beinhalten. Eine Platzierung P ist genau dann eine gültige Lösung S aus dem Raum aller möglichen Lösungen L, wenn alle Elemente von P innerhalb des Rohmaterials R platziert wurden und die Elemente von P sich untereinander nicht überlappen.

$$P = T_1, T_2, T_3, ... | \forall T_i \in 2DKontur \tag{2}$$

$$\forall T_i \in P : T_i \subseteq R \tag{3}$$

$$\forall T_i \in P : \{T_i \cap \{P \setminus T_i\} = \{\}\} \tag{4}$$

$$S \in L | L \text{ erfüllt } (3), (4) \tag{5}$$

Nun ist es im Anwendungsfall nicht nur wichtig, eine existierende Lösung zu finden, sondern diese soll so gestaltet sein, dass möglichst große Teile des Rohmaterials weiterverwendet werden können. Eine Bewertungsfunktion, wie beispielsweise die Fläche der rechteckigen Einhüllenden (F) des verwendeten Materials dient dabei als Zielfunktion. Eine Definition der optimalen Lösung O könnte man folglich so formulieren:

$$\exists O \in S : \forall L \in S : F(O) \leq F(L) \tag{6}$$

2.2 Problemkomplexität

Das Problem lässt sich als Optimierungsproblem beschreiben. Die geometrische Platzierung der Einzelteile ist der zu variierende Parameterraum für die Lösungsbeschreibung. Im Rahmen der Lösungsfindung lassen sich über Berechnungsvorschriften Lösungen iterativ erzeugen, sodass nicht der komplette Parameterraum exploriert werden muss. Hierfür werden dem Parameterraum zusätzliche Konfigurationselemente hinzugefügt.

Die Berechnung der Lösung des allgemeinen Platzierungsproblems für zweidimensionale Rechtecke wurde schon 1981 von Fowler [FoPT81] als sogenanntes NP-vollständiges Problem identifiziert. Auch Hopper stützt sich in seiner Untersuchung [HoTu01] auf diesen Beweis und leitet daraus ab, dass auch die Problemstellung mit komplexeren Objekten in dieser Klasse liegt. Für den hier vorliegenden Fall ist dabei zu beachten, dass die Eingabemenge in ihrer Komplexität eingeschränkt ist. Es handelt sich um Ausgabedaten von CAD Programmen und bilden zu fertigende Objekte ab. Die geometrischen Randbedingungen sind daher mathematisch einfach beschreibbar, besitzen diskrete Stützpunkte und eindeutige Konturen. Gegebene Lösungen sind daher erkennbar und lassen sich eindeutig auf Korrektheit prüfen. Es handelt sich daher nicht um geometrisch uneingeschränkte Platzierungsprobleme, welche deutlich schwieriger zu lösen sind.

In dem vorliegenden Anwendungsfall ist die Eingabemenge aber nicht auf Rechtecke beschränkt, so dass davon ausgegangen werden kann, dass durch die komplexeren Konturen der Objekte weiterhin ein NP-vollständiges Problem vorliegt und sich dieses vermutlich nicht effizient lösen lässt.

2.3 Algorithmen

Zur Problemlösung oder Unterstützung von manuellen Prozessen werden Algorithmen verwendet, um Teilprozesse zu automatisieren und zu beschleunigen. In diesem Abschnitt sollen grundlegende Herangehensweisen an Platzierungsprobleme aufgezeigt und bewährte Algorithmen erläutert werden.

2.3.1 Greedy-Algorithmen Unter dieser Klasse an Algorithmen versteht man Ansätze, bei denen Lösungen für gegebene Problemstellungen schrittweise verbessert werden, um nach einer definierten Anzahl an Schritten oder Erreichen eines stabilen Zustands eine Lösung zu erreichen. Aufgrund der schrittweisen Vorgehensweise, bei der ein Folgeschritt nur zu einer besseren Lösung führen kann, sind Lösungen oft nur lokale Optima. Aufgrund einfacher Optimierungsheuristiken und der sequentiellen Lösungsfindung sind diese Algorithmen aber in den meisten Fällen schnelle Problemlöser.

2.3.2 Evolutionäre Algorithmen Als Problemlöser hat die Natur unzählige beeindruckende Leistungen vollbracht und sich über den Prozess der Evolution an geänderte Randbedingungen angepasst. Diesen Ansatz verfolgen evolutionäre Algorithmen. Sie bilden über stochastische Suchverfahren und Rekombination der besten bekannten Lösungen Evolutionsprozesse ab.[Pohl13]

In diesem Kontext bezeichnet man die Menge der Lösungen als Population. Eine solche besteht aus einzelnen Lösungen, welche aufgrund der Anlehnung an den Evolutionsprozess jeweils als Individuum bezeichnet werden.

Der allgemeine Ablauf des Algorithmus ist wie folgt:

1. Initialisierung mit (zufälligen) Individuen
2. Bewertung der vorliegenden, besten Individuen
3. Selektion von Individuen zur Rekombination
4. Rekombination zu neuen Individuen
5. Mutation - zufällige Variation der Individuen

Die erwähnten, zufälligen Operationen werden im Anwendungsfall meist durch Heuristiken ersetzt, welche den Grad und die Richtung der Änderung einschränken. Zudem werden die Individuen mit Hilfe einer Fitnessfunktion bewertet, welche einen direkten Vergleich mehrerer Lösungen erlaubt und somit eine Zielausrichtung ermöglicht. Je nach gewünschter Näherung der Lösung sind Abbruchkriterien so zu wählen, dass es möglich ist, lokale Optima zu verlassen, um gegebenenfalls eine bessere Lösung zu erreichen. Aufgrund der Heuristiken und des schrittweisen Ansatzes ist jedoch nicht ausgeschlossen, dass die globale Lösung nicht gefunden wird.

2.3.3 Lokale Suche Bei der lokalen Suche wird ähnlich wie bei evolutionären Algorithmen eine Lösung modifiziert, um eine Verbesserung zu erreichen. In der Nachbarschaft einer Lösung wird dabei nach einer besseren Lösung gesucht. Im Gegensatz zu den evolutionären Algorithmen wird aber nur eine Lösung schrittweise verfolgt. Über die Einstellung des Algorithmus ist es möglich, lokale Optima zu verlassen, um bessere Lösungen zu erreichen. Auch sind lokale Optima meist als Lösung eines Problems ausreichend.

2.3.4 Particle Swarm Optimization Ähnlich wie bei einem evolutionären Algorithmus wird eine Population über mehrere Iterationen optimiert. Hierbei wird eine zielgerichtete Mutation verwendet, die die Gesamtmenge der Individuen in Richtung der besten bekannten globalen Lösung modifiziert und so eine breite iterative Annäherung an das Ziel ermöglicht. Es wird eine Richtung und Geschwindigkeit als Optimierungsrichtung pro Individuum berechnet. Die Population bewegt sich wie ein Schwarm auf ein Optimum hin. Auch diese Variante ist anfällig für lokale Optima. Sie eignet sich aber besonders für große Suchräume und mathematisch komplexe Probleme.

2.3.5 Rucksackproblem Im Prinzip handelt es sich bei der gegebenen Aufgabenstellung um eine komplexe Form des in der Informatik oft behandelten sogenannten Rucksackproblems. Bei diesem soll ein endliches Volumen mit einer Teilmenge an Objekten gefüllt werden, welche verschiedene Wertigkeiten besitzen und aufgrund ihres Gesamtvolumens nicht alle in dem Rucksack platziert werden können. Der Unterschied liegt vor allem in der Wertigkeit der Objekte, da die Wertigkeit direkt an die Nutzfläche der Objekte gebunden ist und sich meist untereinander unterscheidet. Zusätzlich ist meist die Eingabemenge kleiner ist als die zur Verfügung stehende Fläche.

2.3.6 Optimierungsproblem über Zielfunktion In der Mathematik werden bestimmte Problemstellungen numerisch gelöst, indem die gewünschte Zielgröße als Funktion definiert wird. Die Parameter dieser Funktion werden dann variiert, um Lösungen zu finden, bei denen die Zielfunktion ihren Maximal- bzw. Minimalwert erreicht. Diese Optimierungen finden in komplexen wissenschaftlichen Feldern An-

wendung, sind aber im Bereich der geometrischen Problemstellungen selten in der rein mathematischen Anwendung vertreten.

2.4 Stand der Wissenschaft

Um einen Überblick über bereits bestehende Ansätze zu erhalten, werden im Folgenden bekannte Arbeiten und Heuristiken vorgestellt. Diese sind nicht zwangsläufig Alternativen, sondern können je nach Architektur teilweise auch kombiniert werden. Die vorhandenen Ansätze sind meist nur Heuristiken, welche eine möglichst gute Lösung, jedoch nicht zwangsläufig die beste aller Lösungen erzeugen können.

2.4.1 BottomLeft Einer der einfacheren und weit verbreiteten Greedy-Ansätze ist ein Verfahren, bei dem neue Teile stets möglichst weit unten links platziert werden. Durch sinnvolles Sortieren der Eingabemenge lassen sich besonders bei rechteckigen Objekten mit geringem Aufwand sehr gute Lösungen generieren. In den meisten Implementierungen beschränkt sich die Verwendung aber auf regulär begrenzte Teile. In verschiedenen Varianten wird die Reihenfolge der Bewegungsrichtung variiert, so dass in bestimmten Fällen die Linksbewegung vor der Bewegung nach unten durchgeführt wird. Auch in neueren Ansätzen wie dem von Siasos [SiVo14] wird dieses Verfahren angewandt.

2.4.2 GRASP Algorithmen Dieser Ansatz hat als Basis einen Greedy-Algorithmus, welcher durch eine lokale Suche ergänzt wird. Die Bezeichnung stammt von der Abkürzung des Terms „greedy randomized adaptive search procedure“ und wird z.B. in der Arbeit von Alvarez[AVPT13] verwendet. Mit dem Ansatz der lokalen Suche lassen sich die einfachen Heuristiken eines Greedy-Algorithmus so ergänzen, dass auch komplexere Strukturen sinnvoll platziert werden können.

2.4.3 NoFitPolygon Neben näherungsweisen und schrittweisen Ansätzen existieren auch geometrisch-rechnerische Ansätze wie das NoFitPolygon. Dabei wird für verschiedene Permutationen eines Objekts rechnerisch ermittelt, an welchen Positionen dieses mit minimaler Lücke zu anderen Teilen platziert werden kann. Als Ergebnis (vgl.

Abbildung 5) werden Linien zurückgegeben, auf denen ein Objekt anhand seines Referenzpunktes platziert werden kann und seine Nachbarn berührt, aber nicht schneidet. Entwickelt wurde das Verfahren schon 1966 [Art 66]. Es wird auch heute noch für Platzierungsprobleme weiterentwickelt[LiHe06]. Eine hohe Eignung besteht bei diesem Ansatz für Probleme mit irregulär begrenzten Flächen.

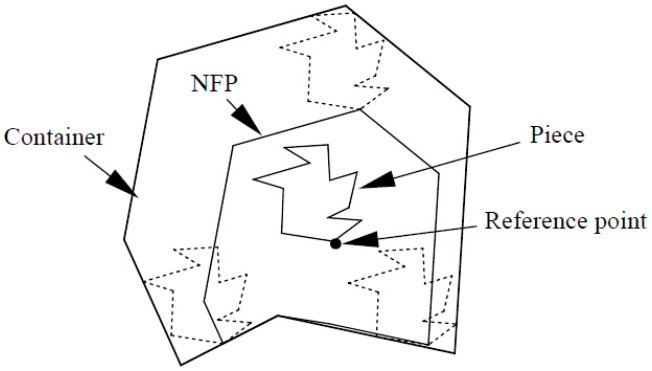

Abbildung 5. Visualisierung eines NoFitPolygon [LiHe06]

2.4.4 Sequenzbasierte Algorithmen Für Platzierungsalgorithmen allgemein üblich und zielführend ist die Variation der Eingangsreihenfolge in Kombination mit einem Platzierungsalgorithmus. Hierbei wird die Eingabemenge permutiert, um eine bessere Lösung zu erreichen. Siasos [SiVo14] zeigt in seiner Arbeit die Leistungsfähigkeit des Ansatzes selbst bei komplexen Geometrien. Große Eingabemengen führen zu einer großen Dimensionalität des Problems, erhöhen aber auch die Möglichkeiten und Varianten um gute Lösungen zu erreichen. Im Umfeld des Prototypenbaus sind die Eingabemengen im Normalfall eher kleiner, sodass untersucht werden sollte, ob sequenzbasierte Algorithmen zur Problemlösung verwendet werden können.

2.4.5 Particle Swarm Optimization Shalaby[ShKa13] nutzt den sequenzbasierten Ansatz und kombiniert diesen mit der Optimierung über Schwarmbewegungen. Er erreicht damit in den üblichen Benchmarks in vielen Fällen die beste bisher bekannte Lösung. Bei

einer in seiner Arbeit durchgeführten Analyse von Einflussfaktoren stellte sich heraus, dass das Verfahren sich besonders gut für Teile mit geringer Komplexität, welche im Verhältnis zum Rohmaterial sehr groß sind, gut eignet. Zurückzuführen auf die Verwendung des sequenzbasierten Ansatzes wird in der Arbeit zudem die große Varianz der Eingabemenge bzgl. der Größe als einflussreicher Faktor für gute Lösungsfindung herausgestellt.

2.4.6 Evolutionäre Algorithmen In Anlehnung an genetische Prozesse hat Jakobs[Jako96] in einer frühen Arbeit einen Algorithmus entwickelt, welcher über umfangreiche Mutationen in Verbindung mit einem einfachen Greedy-Algorithmus erste gute Ergebnisse erzielt hat. Er befasst sich vertiefend mit der erweiterten Mutation von Objekten, wie der Spiegelung oder zielgerichteten Drehung. In neueren Arbeiten, wie der von Thomas[ThCh13], werden meist speziellere Einflüsse, wie die der Fitnessfunktion untersucht. Im Überblick über die Arbeiten der letzten Jahre ist der genetische Ansatz der meist verwendete innerhalb der wissenschaftlichen Veröffentlichungen.

3 Analyse

Vor der Ausarbeitung eines Entwurfs werden in dieser Arbeit sowohl die theoretische Seite der bisherigen Forschungsarbeiten näher betrachtet als auch die praktische Umgebung untersucht. Hierbei soll festgestellt werden, welche Gemeinsamkeiten und Unterschiede zu der Zielsetzung dieser Arbeit bestehen, um mögliche Erkenntnisse daraus zu gewinnen.

3.1 Status Quo

3.1.1 Industrie Im Umfeld der Industrie sind hauptsächlich Großmaschinen in Werkshallen mit ausgebildetem Personal, Schichtbetrieb und hoher Auslastung vorzufinden. Aufgrund der Maschinengröße und der Reduktion von Transportarbeit sind die Prozesse auf lange Bearbeitungszeiten und kurze Rüstzeiten ausgelegt. Als Rohmaterial sind daher folgende Plattengrößen üblich:

Kleinformat	2,00 m x 1,00 m
Mittelformat	2,50 m x 1,25 m
Großformat	3,00 m x 1,50 m
Überformat	6,00 m x 2,00 m

In der Prozessanalyse für industrielle 2D-Trennverfahren ist die wirtschaftliche Betrachtung und Optimierung weit fortgeschritten. Im Allgemeinen kann man hochintegrierte Maschinen mit Einbettung in ein Warenwirtschaftssystem beobachten, die auf Maximierung des Gesamtprofits ausgelegt sind. So werden bestehende Platten oder Rollenmaterial einmalig zur Bearbeitung in die Maschine gelegt und entstehende Reststücke sofort der Schrottsammlung zugeschrieben. Ein erneutes Auflegen von etwaigem, großflächigem Restmaterial ist aus Zeit- und Kostengründen in den Prozessen nicht vorgesehen. Meist wird bewusst mit großem Verschnitt platziert, um ein besseres Handling des Restmaterials zu erreichen, z.B. um die schnelle Entnahme des Schrottmaterials als ein stabiles, zusammenhängendes Teil zu ermöglichen.

Da es bisher kaum etablierte Prozesse und Maschinen für Kleinserienprozesse gibt, werden diese üblicherweise in die Großserienprozesse eingebunden und meist mehrere Kleinserien zu einem Großauftrag

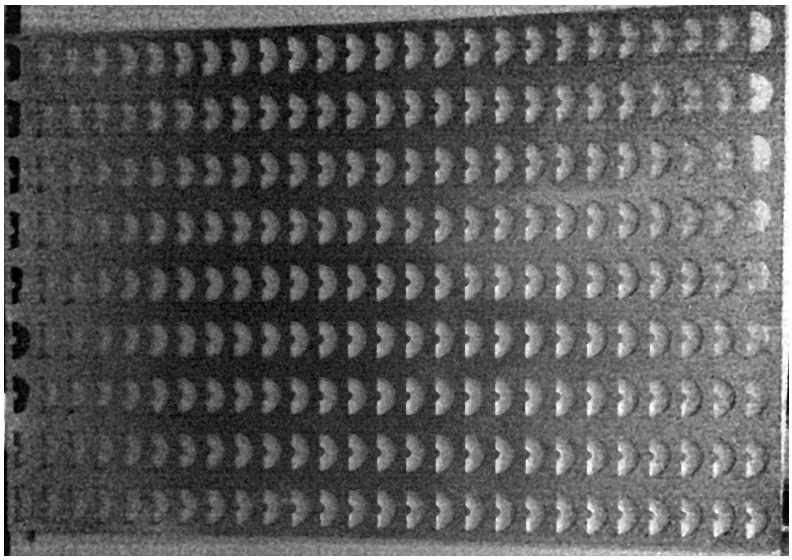

Abbildung 6. Restmaterial einer Kleinserienfertigung aus einem Industrieprozess

zusammengefasst. Dies kann meist nur extern und mit zusätzlicher Zeitverzögerung durchgeführt werden. Im Vordergrund stehen dabei von Seiten des Fertigers hauptsächlich die Gesamtkosten des Prozesses. Neben Maschinen- und Personalkosten sind die Kosten für das Material klein und die Platzierungsoptimierung daher aktuell hauptsächlich für Großserien relevant. Mit zunehmender Automatisierung können aber auch neue Ansätze zu Kosteneinsparungen beitragen und einen Mehrwert liefern. Für deren Entwicklung eignen sich die bereits etablierten Ansätze und Prozesse als Ausgangspunkt.

3.1.2 Rapid Prototyping In der Prototyp- und Kleinserienfertigung sind die Lasersysteme meist als alleinstehende Systeme mit Verbindungen zu Absaug- und Filteranlagen konzipiert. Eine Schnittstelle zum Computer besteht über Netzwerk-, USB- oder Druckerschnittstelle. Die verwendeten Plattengrößen sind wesentlich kleiner als im industriellen Umfeld. Da meist nur wenige Teile zu einem Zeitpunkt aus der Platte geschnitten werden, findet die Positionierung und Anordnung in händischer Arbeit mittels eines Grafikprogramms oder mitgelieferter Gerätesoftware statt. Aufgrund des geringen Ma-

terialdurchsatzes und nur teilweiser Nutzung der Rohmaterialplatten ist es üblich, bereits angeschnittene Platten wiederzuverwenden und auf der noch bestehenden Restfläche neue Teile zu platzieren. Die Platzierung ist daher eher ein Ergänzungsproblem als ein komplettes Packagingproblem. Eine Folge für den Nutzer ist die Lagerung des nun irregulär begrenzten Restmaterials zur späteren Verwendung und Vorhaltung von ausreichendem Rohmaterial.

Abbildung 7. Angebrochenes Material

Die Hersteller kleiner Maschinen bieten nur eingeschränkt Werkzeuge zur Unterstützung. Die mitgelieferten Programme sind meist grafische Benutzerschnittstellen, welche über virtuelle Druckertreiber von Grafikprogrammen mit Fertigungsdaten versorgt werden. Die eingebetteten Funktionen im Bereich der Fertigungsdatenaufbereitung beschränken sich meist auf Pfadsortierung und Anordnung nach rechteckigen Einhüllenden. Die maximale verarbeitbare Komplexität und lange Laufzeiten lassen auf einfachere Lösungsansätze schließen.

Herstellerunabhängige Platzierungssoftware ist, wenn vorhanden, als Plugin für Vektorverarbeitungssoftware (wie z.B. Corel Draw) verfügbar. Aufgrund von Einschränkungen und der mangelnden Qualität wird im Praxiseinsatz oft eine manuelle Platzierung verwendet, bei der die Teile per Maus platziert werden. Dabei wird, um Verschnitt durch optische Ungenauigkeit zu vermeiden, oft viel Sicherheitsabstand zu anderen Teilen und zum Materialrand gelassen. Da bei

wiederholter Verwendung einer Platte diese nicht geradlinig begrenzt ist, ist hier die Konfigurierbarkeit von vielen Platzierungsfunktionen schon überreizt und es bleibt nur die händische Lösung.

Durch die fehlenden Standardprozesse und geringe Automation wird die händische Arbeit mit stark von der Erfahrung abhängiger Qualität und Geschwindigkeit durchgeführt. Als Ziel ist hier die einfache Handhabung des Materials und gute Materialausnutzung deutlich im Vordergrund, wobei dies durch kleinere Verzögerungen beim Bestücken der Maschine oder Vorabberechnungen geschehen kann.

Eine Teilautomatisierung der bisherigen händischen Prozesse wäre für die Anwendung ein großer Zugewinn und würde sowohl eine Zeitersparnis als auch eine effektivere Materialnutzung mit sich bringen. Eine Gemeinsamkeit zu den klassischen Großserien ist gegeben, wenn auch mit anderen Prioritäten und Randbedingungen.

3.1.3 Wissenschaft In der Forschung findet die Problemstellung des Platzierens an vielen Universitäten großes Interesse und liefert viele verschiedene Testfälle und Lösungsansätze. Diese sind meist komplett auf theoretischer Ebene angesetzt und behandeln hauptsächlich generierte Datensätze einer charakteristisch sehr gleichbleibenden Eingangsmenge. Der Lösungsansatz ist daher in vielen Fällen speziell für den jeweiligen Anwendungsfall angepasst. Beispielsweise behandelt Schöning [STTM$^+$01] in seiner Arbeit die Anordnung von Fadenspulen als Platzierungsproblem. Pinheiro[PiAJS15] behandelt Benchmarkdatensätze aus der Textilindustrie, während sich Goncalves [GoRe13] mit reinen rechteckigen Objekten beschäftigt.

Hierbei steht es im Vordergrund eine Lösung nahe des theoretischen Optimums zu finden und Laufzeituntersuchungen werden nebensächlich behandelt. Um verschiedene Ansätze zu vergleichen haben sich aus den älteren Arbeiten mittlerweile mehrere Datensätze als Benchmarks etabliert. Die Special Interest Group on Cutting and Packaging[oCPa15] hat diese gesammelt und stellt sie aufbereitet zur Verfügung. Für einen Vergleich mit anderen Arbeiten finden mehrere Datensätze auch in dieser Arbeit Verwendung.

Trotz der meist eher theoretischen Umsetzungen der wissenschaftlichen Arbeiten besteht eine große Aussagekraft und exakte Untersuchung von Einzelfällen. Diese können in den praktischen An-

wendungsfällen direkt identifiziert und passend gelöst werden. Eine direkte Anwendbarkeit ist nicht möglich, da sehr viele Annahmen und Ausnahmeregelungen bestehen, welche in Arbeitsprozessen so nicht umsetzbar sind. Vor allem die allgemeinen Lösungsansätze bieten hohe Gemeinsamkeiten mit etablierten Verfahren verschiedenster Anwendungsfälle.

3.1.4 Sonstige Ein Bereich der 2D-Trennverfahren, der aktuell ein sehr starkes Wachstum zu verzeichnen hat, ist der Heimbastelbedarf und Modellbau. In diesem Bereich werden vor allem einfache Geometrien geschnitten, wobei die Varianz der Teile wesentlich stärker ist als im industriellen Umfeld. Neben dem reinen Materialschnitt werden im gleichen Arbeitsgang oft auch Gravuren durchgeführt. Für das Packaging gilt es zu beachten, welche Konturen dabei relevant sind. Aufgrund des privaten Umfelds ist das Zielkriterium in fast allen Fällen die bestmögliche Ausnutzung des vorhandenen Materials bei minimalem Verschnitt. Die dafür investierte Arbeitszeit, sowohl beim Rüsten der Maschine als auch der Berechnung der Platzierung, wird dabei in Kauf genommen. Die grundsätzliche Problematik ist daher gleich der in der Großindustrie, zeitliche Faktoren haben aber weniger negativen Einfluss auf die Bewertung einer Praxistauglichkeit. Dennoch sind hier automatisierte Prozesse aus Nutzer- und Herstellersicht willkommen.

3.1.5 Herausforderungen Beim Entwurf von Platzierungsalgorithmen für das Rapid Prototyping durch 2D-Trennverfahren gilt es die verschiedenen Kriterien unterschiedlicher Verfahren und Zielsetzungen zu berücksichtigen. Zusätzlich variieren die Anforderungen an Genauigkeit, Geschwindigkeit und Kosten sowie die Charakteristik der zu fertigenden Bauteile zwischen den Branchen.

Eine teilautomatisierte Lösung, die die bestehenden Prozesse unterstützt und die manuelle Nestingarbeit übernimmt, muss also unter verschiedenen Randbedingungen funktionieren und sich auch auf verwandte Fertigungsverfahren anwenden lassen.

3.2 Folgerungen

Bei der Analyse der bestehenden Ansätze fällt trotz der verschiedenen Anwendungsgebiete auf, dass große Gemeinsamkeiten bestehen, wobei die Eignung im Speziellen eine hohe Abhängigkeit von der Eingabemenge besitzt. Alle verwendeten Ansätze haben sich über lange Zeiträume bewährt und liefern in Teildisziplinen Spitzenwerte, sodass keine direkte Tendenz des erfolgversprechendsten Ansatzes vorliegt.

Für die Anwendung selbst ist im Prototyping-Umfeld die Materialausnutzung und sinnvolle, wiederholte Verwendbarkeit deutlich im Vordergrund. Die Laufzeit ist dabei nicht primäre Zielgröße, solange die Berechnung im späteren Prozess keine Zeitunterbrechung nach sich zieht.

Bei der Sichtung der Literatur hat sich die geringe Komplexität von Eingabeteilen als unerwartet herausgestellt. Ganz offensichtlich, aber auch teilweise explizit herausgearbeitet (vgl. [ShKa13]) ist dies ein ausschlaggebender Berechnungsfaktor. Um eine praxistaugliche Lösung bieten zu können, werden im Rahmen dieser Arbeit folgende Einschränkungen an die Eingabemenge gefordert:

- Es gibt mehrere gültige Lösungen (kein PerfectFit)
- Alle Objekte sollen in einem Fertigungsschritt getrennt werden
- Die Geometrien haben eine geschlossene, unverzweigte Kontur (z.B. keine Stichfahrten)
- Löcher in Teilen müssen nicht verwertet werden (d.h. großflächige Ausschnitte zählen nicht als Restmaterial)
- Die Anzahl an Objekten pro Eingabemenge ist im Kleinserienbereich (< 20 Teile)

4 Entwurf

4.1 Vorüberlegungen

Sowohl der Vergleich der vorliegenden wissenschaftlichen Arbeiten als auch die nähere Betrachtung der Anwendungsfälle zeigt deutlich, dass ein Berechnungskonzept sehr allgemein gehalten sein muss und mit verschiedenen Charakteristika der Eingabedaten zurecht kommen muss.

4.2 Ansätze

In Anlehnung an die vorgestellten wissenschaftlichen Arbeiten und den aktuellen Stand der Technik werden mehrere Ansätze und Teilideen vorgestellt, deren Verwendbarkeit untersucht wird.

4.2.1 Coastline Inspiriert von den manuellen Heuristiken, welche Heistermann[HeLe95] bei der Platzierung auf Tierfellen durch Gerber (siehe Abb. 8) beobachtet hat, kann über die Implementierung des No-Fit-Polygon-Algorithmus für jedes Teil die Grenze des Bewegungsraum aufgezeigt werden, in dem es ohne Kollision platziert werden kann. So kann der Raum möglicher Lösungen auf einen Bruchteil reduziert werden. Aus diesen bekannten Randpositionen kann die Position mit dem geringsten Verschnitt effizient berechnet werden, da diese sich in einem der Eckpunkte befindet. Die meisten Problemstellungen besitzen eine Optimierungsrichtung, weshalb das Verfahren vergleichbar ist mit dem Begradigen einer Küstenlinie durch eingefügte Konturen.

4.2.2 Mathematische Differenz Wenn die Kanten von Objekten als mathematische Funktionen beschrieben werden, lassen sich besonders platzsparende Platzierungen finden, indem die Fläche zwischen den Funktionen als Wertfunktion betrachtet wird. Durch eine passende Intervall- und Integralberechnung lassen sich so die besten Ergänzungen für Lücken finden.

4.2.3 Klassifikation Um bei der Platzierung auf passendere Heuristiken zurückgreifen zu können, lassen sich Eingabeobjekte nach

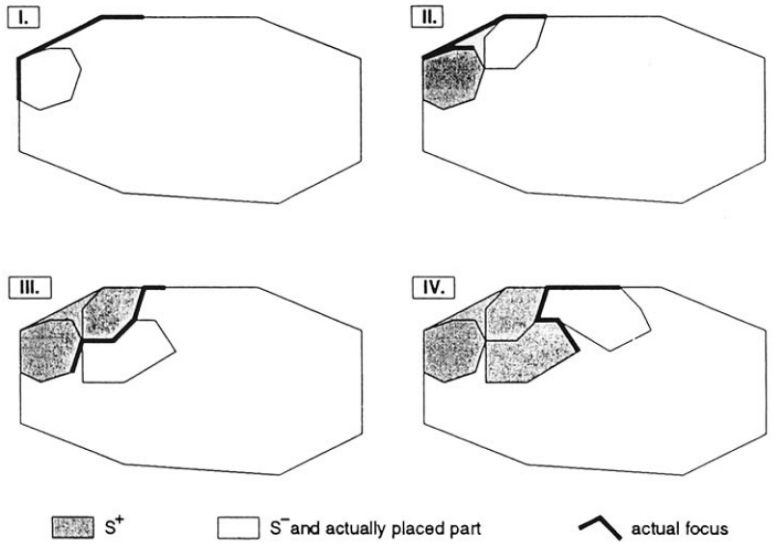

Abbildung 8. Anordnung am Rand des Materials [HeLe95]

ihren Charakteristika klassifizieren und so effizienter anordnen. Speziell über die Kenngröße des Füllgrads und Ähnlichkeiten mit der Kreis- oder Rechtecksform können für die Platzierungsreihenfolge wertvolle Vorsortierungen stattfinden.

4.2.4 Permutation Ein gängiger Ansatz für die Verbesserung von Greedy-Sortieralgorithmen ist die Permutation der Eingabemenge. Es wird dabei meist durch ein Vertauschen der Reihenfolge eine bessere Gesamtplatzierung erreicht. Da die Eingabemengen üblicherweise viele Elemente enthalten, ist großes Verbesserungspotential vorhanden. Durch Parallelisierung können besonders viele Permutationen generiert, platziert und bewertet werden.

Im Bereich der Kleinserien- und Prototypenfertigung ist die Anzahl der Elemente in der Eingabemenge meist klein und damit die Anzahl der Reihenfolgenpermutationen deutlich beschränkt. Neben der Permutation der Reihenfolge können die Objekte aber auch in anderen Aspekten permutiert werden. Für die Platzierung kann z.B. eine Rotation durchgeführt werden. Um die Anzahl der Permutationen bei Winkeln zu reduzieren, wird üblicherweise eine Schrittweite vorgegeben. Auch die Spiegelung kann pro Objekt bis zu drei zusätz-

liche gültige Permutationen erzeugen und den Lösungsraum damit maßgeblich erweitern.[Jako96] Da die Spiegelung in vielen Anwendungsfällen aufgrund des Materials nicht möglich ist und auch in Benchmarks kaum vorkommt, wird diese Option im Rahmen der Arbeit nicht genutzt.

4.2.5 ParticleSwarm Der ParticleSwarm-Ansatz wurde in Arbeiten im Aufgabengebiet des Packaging bereits verwendet. Shalaby [ShKa13] zeigt in seiner Arbeit eindrücklich die Leistung des Verfahrens. In der Optimierungsphase wird dabei die Reihenfolge der Eingabeteile variiert, um die beste Lösung zu erhalten. Über einfache Mutationsoperatoren wird die Menge zwischen den Durchgängen verändert. Durch eine Fitnessfunktion werden dann die besten Lösungen weiter iterativ mutiert.

4.2.6 Evolutionärer Ansatz Als mächtige Algorithmenstruktur bietet sich der evolutionäre Ansatz an. Es können im Gegensatz zu den rein sequentiellen Ansätzen auch parallel Lösungen gesucht und eine Menge an Lösungen der letzten Berechnungsgeneration angeboten werden. Aufgrund der vielen unterschiedlichen Möglichkeiten, die geforderten Objekte auf einem Medium zu platzieren. ist die Definition und Berechnung der Zielfunktion nicht einfach. Ein evolutionärer Ansatz funktioniert jedoch auch mit einer komplexen Zielfunktion, da er für die iterative Lösungsfindung keine genauen Information oder gar Ableitungen der Zielfunktion benötigt. [Pohl13]

Die für den evolutionären Algorithmus benötigten Komponenten (vgl. Abbildung 9)

- Selektion,
- Rekombination,
- Mutation,
- Bewertung und
- Wiedereinfügen

sind in vielen Varianten und Zielsetzungen in der Fachliteratur und weiteren Veröffentlichungen dokumentiert, sodass eine für den Einsatzzweck geeignete Zusammenstellung erarbeitet werden kann. Hierbei gilt es zu beachten, dass die Bewertung einer Lösung in diesem Kontext auch eine aufwändige Berechnung beinhalten kann. Im Falle

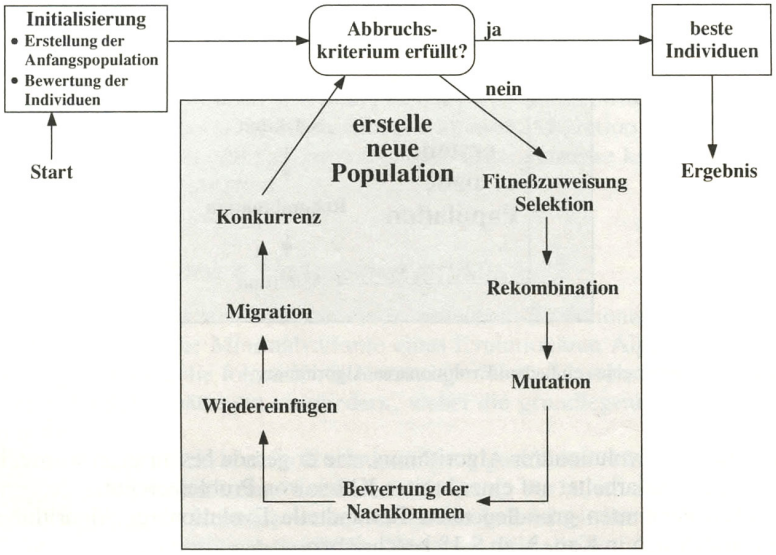

Abbildung 9. Ablauf eines evolutionären Algorithmus [Pohl13]

einiger Arbeiten wird z.B. eine Platzierung aller Teile über einen Greedy-Algorithmus vorgenommen und diese berechnete Lösung dann bewertet.

Wenn zur Lösung der Aufgabenstellung ein Ansatz über Reihenfolgenoptimierung gewählt wird, spricht man im Rahmen der Optimierung von einer kombinatorischen Problemstellung. D.h. es wird eine Permutation der Liste von zu platzierenden Teilen gesucht, die mit einer einfachen Platzierungsvorschrift zur besten Lösung führt. Um bei kombinatorischen Problemen vielversprechende Mutationen und Rekombinationen zu erzeugen wird daher auf die Nachbarschaftserhaltung geachtet. Bestehende Reihenfolgen sollen also stückweise erhalten bleiben, um die Chance zu erhalten, dass mehrere gute Teilsequenzen zu einer besseren Gesamtlösung führen.

4.2.7 Lokale Suche Die lokale Suche eignet sich, um bereits bestehende Lösungen und Teillösungen zu verbessern. Eine bestehende Lösung wird dabei sequentiell durch Modifikationen weiterentwickelt und dabei der Wert der Zielfunktion verbessert. Als alleinstehender Ansatz ist diese Methode aufgrund der großen Gefahr von lokalen Optima weniger geeignet, kann aber auch im Rahmen eines anderen

Ansatzes als Teillösung Anwendung finden. Folgende Ansätze können für Platzierungsprobleme verwendet werden:

Zufallsverschiebung: Ein Beispiel der lokalen Suche wurde von Schöning [STTM+01] implementiert. Es wurden in der Arbeit für eine Logistikanwendung die Packungsdichten von Garnspulen untersucht und verbessert. Über ein zufälliges Rütteln und Neubegrenzung der begrenzenden Kontur konnten iterativ dichte Packungen für verschiedene Spulendurchmesser gefunden werden. (siehe Abb.10)

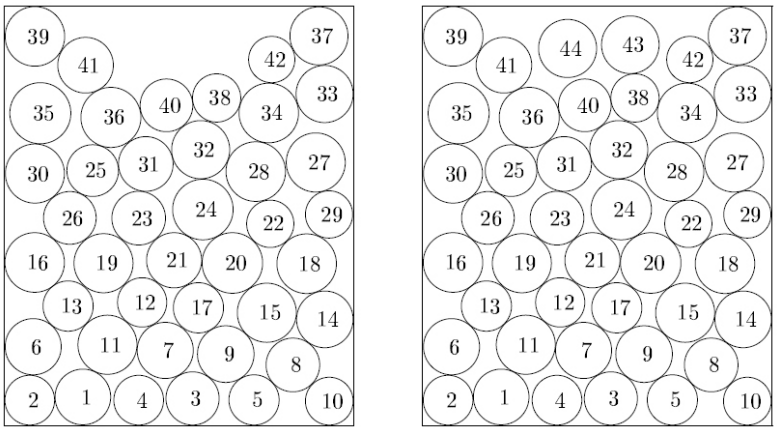

Abbildung 10. Rütteln von Garnspulen zur Verkleinerung des Transportvolumens[STTM+01]

Kräftesimulation: Da es in vielen Fällen um lückenfreies Platzieren am unteren Materialrand geht, liegt es nahe, auch in verwandten, virtuellen Welten nach Lösungen zu suchen. Im Bereich der Materialwissenschaft, Physik und Computerspielbranche sind zunehmend auch annähernd genaue Physiksimulationen verfügbar. Als Anpassung auf die Problemstellung kann eine vereinfachte zweidimensionale Physiksimulation genutzt werden um Objekte aufeinander zu platzieren. Algorithmen wie Bottom-Left können näherungsweise durch verschiedene Schwerkraftsvektoren abgebildet werden. Der Vorteil dieser Berechnung ist die Interaktion zwischen bereits platzierten Teilen und den Neueingefügten. Es können dabei kleine Verschiebungen

und Drehungen stattfinden, die auf analytischem Wege kaum sinnvoll zu berechnen sind.

Reihenfolge: Statt der Variation der direkten geometrischen Anordnung kann auch die Einfügereihenfolge variiert werden. Shalaby[ShKa13] stellt in seiner Arbeit drei einfache Mutationsvorschriften vor, welche die Reihenfolge zufällig verändern. Über eine mehrfache Anwendung und Auswertung können so iterativ Lösungen verbessert werden. Der Einfluss der Einfügereihenfolge ist von großer Bedeutung, da die Einfügealgorithmen stets iterativ versuchen, eine beste Teillösung zu generieren, und durch eine ungünstige Reihenfolge keine gleichwertigen Lösungen erreicht werden können. Eine hohe Abhängigkeit besteht daher auch von der Diversität der Teile. Handelt es sich um viele Gleichteile sind Tauschoperationen je nach betroffenen Teilen unterschiedlich wirksam. Ähnliche Effekte gelten auch für die allgemeinen Maße und Charakteristika der Teile. Da viele Heuristiken bereits bekannt sind, können in einigen Fällen durch intelligente Vorsortierung schnell gute Lösungen erzielt werden. Für einen Platzierungsalgorithmus muss in jedem Falle eine irgendwie geartete Berücksichtigung und Anpassung der Reihenfolge stattfinden.

4.3 Modelldarstellung

Zur Lösung des Platzierungsproblems ist nicht nur eine Berechnungsvorschrift nötig, es sind auch begleitende Konzeptentscheidungen zu fällen, die sich mit der Datenmodellierung befassen.

4.3.1 Konturen Um Objekte geometrisch beschreiben zu können, soll im Rahmen der Arbeit eine vektorbasierte Modellierung verwendet werden, die eine exakte Repräsentation erlaubt. Da Eingabedaten in Vektordatenformaten vorliegen, können einfache Übertragungen stattfinden. Die Objekte werden in Anlehnung an die Spezifikation der Scalable Vector Graphics des World Wide Web Consortium (W3C) modelliert. Daher wird die Kontur eines Objektes als geschlossener Pfad mit mehreren Pfadstücken repräsentiert.

Als Pfadstücke stehen dabei folgende Möglichkeiten zur Verfügung:

- Sprung zu einem definierten Punkt - ohne Verbindungslinie
- Gerade vom aktuellen Punkt zu einem weiteren Punkt

– Kurve vom aktuellen Punkt zu einem neuen Punkt über zwei Hilfspunkte

Die Position der Stützvektoren wird dabei in einem globalen Koordinatensystem gespeichert.

4.3.2 Verschachtelung Es ist nötig, zusätzlich zu den Außenkonturen auch Innenkonturen darstellen zu können. Zu diesem Zweck ist es möglich, Konturen einer Geometrie zuzuordnen, die innerhalb der Außenkontur liegen. Diese können nun als Objektgruppen verarbeitet werden. So können Hierarchien aufgebaut werden, die für weitere Operationen wie Kollisionsabfragen und Bewertungsfunktionen verwendet werden können.

4.3.3 Kollisionsabfrage Um die Korrektheit einer Lösung zu gewährleisten, muss während der Berechnung, zur Bewertung und zur Kontrolle vor dem Export eine Kollisionsprüfung stattfinden. Da sich diese bei einer großen Anzahl an Objekten als Elemente einer Lösung als komplex erweist, werden verschiedene aus der Computergrafik bekannten Hierarchien verglichen, die eine Verbesserung der Berechnungslaufzeiten ermöglichen.

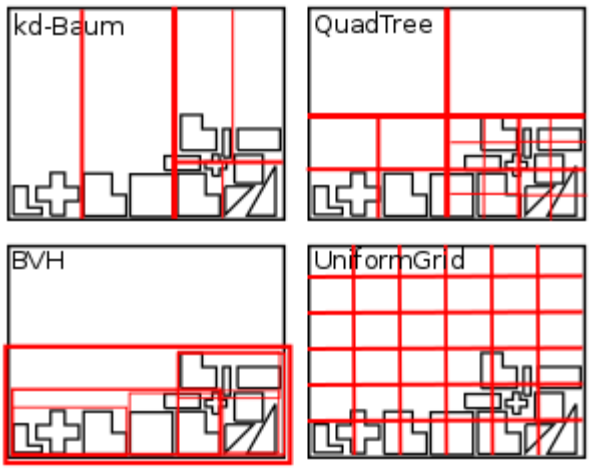

Abbildung 11. Verschiedene Möglichkeiten der Repräsentation

BSP-Baum / kd-Baum: Der zur Verfügung stehende Raum wird durch Trenngeraden geometrisch geteilt. Für eine Kollisionsabfrage kann schrittweise festgestellt werden, auf welcher Seite einer Trenngeraden das aktuelle Objekt liegt und welche weiteren Objekte dort verzeichnet sind. Zur einfachen Berechnung werden in der Variante des kd-Baumes nur zum Koordinatensystem parallele Trennlinien verwendet, beim BSP-Baum können diese beliebig platziert werden. Je nach Implementierung sind die Einfügeoperationen und Balancingvorschriften unterschiedlich implementiert, sodass bis zu einem bestimmten Grad auch unbalancierte Bäume genutzt werden können.

Vorteile	Nachteile
Einfache Baumerzeugung	Häufige Neuberechnung nötig
Kann auch unbalanciert genutzt werden	Aufwand bei der Traveriserung

QuadTree: Ähnlich wie beim kd-Baum wird der Raum achsensymmetrisch unterteilt. Beim QuadTree ist die Position der Trennlinien bereits definiert. Jede Unterteilung trennt den (Unter-)Raum genau in vier gleich große Regionen. Dies ermöglicht eine rechnerisch einfache Einteilung in Unterräume und ermöglicht in allen Unterräumen eine überschaubare Anzahl an Objekten zu halten. In der zu lösenden Kollisionsabfrage bietet QuadTree die Möglichkeit, die Regionen eines Rohmaterials genau abzubilden, in denen viele Objekte platziert wurden, und kann so die Anzahl der zu prüfenden Objekte reduzieren.

Vorteile	Nachteile
Bildet Häufungen gut ab	Häufige Neuberechnung nötig
Partielle Neuberechnung möglich	Aufwand bei der Traveriserung

Bounding Volume Hierachy: Auch die BVH hat das Ziel ungleich verteilte Anordnungen so zu repräsentieren, dass eine Zuordnung in überschaubare Unterräume möglich ist und effizient geprüft werden kann, ob Überlappungen entstehen. Es wird eine Baumstruktur erstellt, bei der jeder Knoten die geometrische Einhüllende aller Kindsknoten beinhaltet und so über einfache Abfragen geklärt werden kann, in welchem Kindsknoten sich ein Objekt befindet und welche mögliche Nachbarobjekte sind. Die Erstellung und Ausbalancierung des

Baumes ist in diesem Anwendungsfall aufwändig, da viele Einfügeoperationen beim Platzierungsvorgang durchgeführt werden und durch die irregulären Konturen der Teile meist keine überlappungsfreien, einhüllenden Geometrien gefunden werden können.

Vorteile	Nachteile
unausbalanciert / überlappend möglich	Häufige Neuberechnung nötig Ungünstig bei iterativem Aufbau Aufwand bei der Traveriserung

Uniform Grid: Als statische Einteilung wird beim Uniform Grid die komplette Arbeitsfläche regelmäßig unterteilt und zu jedem Feld eine Liste aller berührenden oder beinhalteten Objekte geführt. Dies bietet einen schnellen Zugriff und effiziente Einfügeoperationen. Aufgrund seiner festen Einteilung besteht bezüglich der Leistungsfähigkeit und Effizienz eine hohe Abhängigkeit von der geeigneten Wahl der Zellgrößen. Die Repräsentation eignet sich hinsichtlich Speicherplatzeffizienz schlecht, wenn Objekte im Raum nicht annähernd gleichverteilt sind.

Vorteile	Nachteile
Schneller Zugriff	Aufwändiger initialer Aufbau
Gute Abbildung von konkaven Objekten	Performance abhängig von Gridparametern
Direkter Zugriff	
Nur lokale Änderungen	

Bewertung: Nach genauer Betrachtung der verschiedenen Repräsentationen der Anordnung zur effektiven Kollisionsabfrage wird im weiteren Verlauf die Uniform-Grid-Variante als geeignetster Ansatz gewählt. Durch die iterative Vorgehensweise der Algorithmen sind viele Einfüge- und Änderungsoperationen zu erwarten, welche bei den anderen Lösungen zu zusätzlichem Reorganisationsaufwand führen können. Zudem entfällt der Aufwand der Traversierung, da über die bekannte Kontur und Position eines Objektes direkt auf die relevanten Zellen (und damit auf die Nachbarobjekte) zugegriffen werden kann.

4.4 Algorithmenentwurf

Für die Umsetzung im Rahmen dieser Arbeit wird nun aus den genannten Ansätzen ein vielversprechendes Konzept zur Lösung der Platzierungsaufgabe gewählt und ausgearbeitet.

4.4.1 Vorabüberlegungen Für die Konzeption eines geeigneten Ansatzes müssen einige grundlegende Entscheidungen getroffen werden, um eine zielgerichtete Entwicklung zu erlauben.

Geforderte Genauigkeit und Zielerreichung: Unter näherer Betrachtung gibt es Verfahren, die, mit erhöhtem Aufwand, eine exakte bzw. optimale Lösung anstreben und Verfahren, die über Näherungen und Heuristiken schnell gute Lösungen erzeugen. Im Rahmen der Anwendung ist eine hohe Materialersparnis das Ziel, aber aufgrund der direkten Einbindung in den Arbeitsprozess durch die zur Verfügung stehende Berechnungszeit beschränkt. Die aktuell verwendeten generischen Ansätze sind bereits ausreichend gut um Kosteneinsparungen zu bewirken. Eine Lösung, die für Sonderfälle die bestmögliche Anordnung findet, hat daher wirtschaftlich nur beschränkt großen Mehrwert. Die sogenannten PerfectFit-Problematiken sind eine Problemstellung, welche in der Praxis aber extrem selten auftreten. Eine Lösung über Approximationen und einfache Optimierungsvorschriften ist zur Platzierung daher im Regelfall ausreichend.

Parallelität: Je nach Algorithmus und Implementierung sind einige Konzepte auf hohe Parallelität ausgerichtet bzw. können ihre Vorteile auf spezieller Hardware wie GPUs oder Rechnerclustern ausspielen. Für den betrachteten Anwendungsfall ist eine Anwendung auf allgemeinen Endgeräten wie Laptops oder Desktoprechner üblich und damit keine spezialisierte Hardware vorhanden. Die Auswahl eines Algorithmus richtet sich daher auf sequentielle Verfahren oder einfache Parallelisierung.

Evolutionäre und Schwarmansätze: Viele der betrachteten Ansätze nutzen die Permutation der Einfügereihenfolge, um gute Platzierungen zu erreichen. Dies ist besonders bei vielen Objekten ein effektiver Ansatz. Die im Rahmen der Arbeit betrachteten Anwendungsfälle haben klassischerweise kleine Eingabemengen und bieten daher nur

einen beschränkten Permutationsraum für Mutationen. Dieser Raum kann durch weitere Objektparameter wie Rotationen und Startpositionen erweitert werden, um auch bei kleiner Objektanzahl viele Mutationsmöglichkeiten zu liefern und eine Eignung für evolutionäre Ansätze zu bieten. Durch die Erweiterung der Dimension der Parameter ist die Implementierung eines Schwarmalgorithmus wegen der vieldimensionalen Schwarmbewegung aber deutlich erschwert. Eine Umsetzung als klassischer evolutionärer Algorithmus dagegen ermöglicht einen hohen Flexibilitätsgrad, Parametrierbarkeit und passende Berechnungskomplexität für diesen Anwendungsfall.

Lokale Suche: Für viele Ansätze der lokalen Suche wird schnell deutlich, dass eine starke Abhängigkeit von Objektgeometrien besteht, um wirklich effizient Lösungen erzeugen zu können. Für eine gute Eignung im Anwendungsfall muss also eine Klassifizierung der Eingabemenge durchgeführt und darauf passende lokale Suchen angewendet werden. Da fast nur fiktive Datensätze zur Verfügung stehen und die wenigen Realdaten keine Rückschlüsse auf die Charakteristika und Auftretensverteilungen der Objektgeometrien zulassen, ist eine praxisnahe Umsetzung nur über grobe Klassifizierung möglich. Im Rahmen dieser Arbeit wird daher kein Fokus auf die direkte Lokale Suche gelegt, sondern die Heuristiken an erkannte Charakteristika angepasst.

In Anlehnung an die zielgerichtete Suche soll im Rahmen der Arbeit untersucht werden, ob zufällige Anteile der Optimierungsoperationen ergänzt und ausgerichtet werden können. Durch die Vorkenntnissen über die Optimierungsproblematik und die verwendeten Platzierungsverfahren könnte Verbesserungspotential genutzt werden.

4.4.2 Struktur Auf Basis der vorgestellten Vorüberlegungen wird ein evolutionärer Algorithmus entworfen, welcher sich über verschiedene Komponenten auf eine möglichst große Breite an Aufgabenstellungen im Umfeld der Prototypen- und Kleinserienfertigung anwenden lässt.

Listing 1. Evolutionärer Algorithmus als Pseudocode

```
createInitialPopulation
while(abortionCriteria.notFulfilled)
        selectTopIndividuals
        foreach(topIndividual)
                recombinate(topindiv, randomIndiv)
        foreach(individuals)
                foreach(mutation)
                        mutate(individual)
                        newIndiv.add(mutatedIndividual)
        foreach(newIndividual)
                greedy.package(newIndividual)
        sort(newIndividuals)
        truncate(newIndividuals)
        truncate(lastPopulation).concat(newIndividuals)
return greedy.package(bestIndividual)
```

4.4.3 Platzierung Da der Ansatz der Optimierung über die evolutionäre Permutation der Eingabemenge gewählt wurde, ist es nötig, eine einfache, reproduzierbare Einfügevorschrift umzusetzen, mit der eine Platzierung und damit eine Bewertung stattfinden kann. Hierzu kann ein einfacher Greedy-Algorithmus eingesetzt werden. In Anlehnung an andere wissenschaftliche Arbeiten wird der schon seit Hopper [HoTu99] bewährte Bottom-Left-Algorithmus umgesetzt. Hierbei wird ein neu eingefügtes Objekt von oben auf die bereits gesetzte Platzierung geführt. Sobald dieses an einem anderen Objekt oder dem Rand des Rohmaterials anstößt, wird die Abwärtsbewegung gestoppt und eine Linksbewegung durchgeführt. Wenn diese durch ein anderes Objekt oder den Materialrand gestoppt wird, kann man die beiden Schritte so lange wiederholen, bis eine stabile Position erreicht ist, d.h. keine Verschiebung nach unten oder links mehr möglich ist. Dann kann das nächste Objekt auf die gleiche Weise platziert werden.

Listing 2. Platzierungsalgorithmus als Pseudocode

```
foreach(sequenceOfParts)
        setToInitialPlacement(part)
        while(!isStable(part))
                slideBinary(part, bottom)
                slideBinary(part, left)
        if(isValidPosition(part))
                canvas.add(part)
        else
                canvas.setInvalid
return canvas

Function slideBinary(part)
        while delta > threshold
                part.MoveBy(delta)
                        if(!isValidPosition())
                                part.MoveBy(-delta)
                 delta = delta / 2
```

Wie genau die einzelnen Komponenten umgesetzt werden sollten, wird im Folgenden einzeln erläutert.

4.4.4 Individuen und Populationen Für den gewählten evolutionären Ansatz ist neben der Wahl der einzelnen, berechnenden Algorithmenkomponenten auch die Ausprägung der Populationen und Individuen entscheidend. Hierzu werden die sogenannten Chromosomen eines Individuums definiert, welche im Verlauf der Berechnung mutiert werden können und ausreichenden Einfluss auf die Güte einer Lösung haben. Es wurden folgende Chromosomen pro Individuum definiert:

- Reihenfolge der Elemente
- Startposition pro Element
- Startrotation pro Element

Die gesamte Lösungsmenge wird als Population bezeichnet, während einzelne Lösungen Individuen sind. Die Gesamtgröße einer Population ist über Simulationsparameter begrenzt und wird über die Wiedereinfügevorschrift unter der gegebenen Schranke gehalten.

	Ursprungsgeometrie	Startposition	Startrotation
Geometrie 1	13	1|4	15°
Geometrie 2	3	3|2	0°
Geometrie 3	1	5|4	180°
⋮	⋮	⋮	⋮
Geometrie n	8	4|8	30°

Abbildung 12. Beispielausprägung eines Individuums

4.4.5 Bewertung Die Bewertung eines Individuums findet über die Verwendung seiner Chromosomen statt. In diesem Falle wird die Ausprägung von Einfügereihenfolge, Startposition und Startrotation von einem Greedy-Algorithmus auf das Rohmaterial platziert und diese Lösung bewertet. Nach welchen Kriterien die Bewertung genau stattfindet, wird in einem späteren Kapitel (siehe Kapitel 6.4) behandelt. Zu betonen ist, dass die Bewertung in der entworfenen Struktur einen großen Berechnungsaufwand und den eigentlichen Platzierungsvorgang beinhaltet.

4.4.6 Selektion Für die Rekombination von Individuen ist es nötig, jeweils die vielversprechenden Paarungen auszuwählen. In diesem Selektionsprozess werden nach einem vorgegebenen Anteil die besten Individuen einer Generation mit einer zufälligen Auswahl aus der Restmenge gekreuzt. Dies ermöglicht es, die Paarungen nicht beidseitig elitär zu gestalten und zu schnell in einen Verlust der Diversität von Ausprägungen zu gelangen.

4.4.7 Rekombination Die bestehende Population wird im Rahmen der Rekombination durch Kreuzung verschiedener Individuen erweitert. Bereits bewährte Literaturansätze können übernommen und angepasst werden. Um aus guten, bestehenden Lösungen mit möglichst hoher Wahrscheinlichkeit auch gute Nachkommen zu generieren, ist es wichtig, Ansätze zu wählen, bei denen Zusammenhänge erhalten bleiben können. Da zum Beispiel die Einfügereihenfolge einen großen Einfluss auf die Güte einer Lösung hat, ist es erstrebenswert Rekombinationen zu wählen, die stückweise nachbarschaftserhaltend sind.

Für die Rekombination ist es allgemein von Interesse eine hohe Diversität der Individuen zu besitzen, um die Möglichkeit zu gewährleisten, bei einer Rekombination die besten Ausprägungen beider Eltern zu einer noch besseren Lösung zu kombinieren. Auf der Gegenseite sollten die Individuen der Ausgangspopulation schon relativ gute Lösungen darstellen, um die Chance für unbrauchbare neue Lösungen gering zu halten. Um zu verhindern, dass im späteren Verlauf einer Durchführung nahezu alle Individuen nur geringfügige Abweichungen der gleichen Lösung darstellen und Rekombinationen wenig Mehrwert bieten, sind die oben genannten Parameter entsprechend zu bestimmen. Zusätzlich kann bei manchen Rekombinationsoperationen durch eine Rekombinationsrate angegeben werden wie groß der vererbte Anteil eines Elternteils ist.

Crossover: Klassisch wird bei der Rekombination zweier Eltern der Crossoveroperator verwendet. Dabei werden beide Eltern an den gleichen, zufällig gewählten Stellen getrennt und der Nachkomme übernimmt abwechselnd einen Teil von jedem Elternteil. Je nach Ausführung kann dies mit einer Trennstelle (singlepoint crossover) oder an mehreren Stellen (multipoint crossover) durchgeführt werden. Da sich dieser Operator in seiner Ursprungsform nur auf binäre Ausprägungen anwenden lässt, ist er für den kombinatorischen Anwendungsfall nicht geeignet.

Position Crossover: Für kombinatorische Probleme werden häufig Position Crossover Rekombinationen verwendet. Aus dem ersten Elternteil wird eine definierte Anzahl an Positionen zufällig ausgewählt. Diese werden im Kind an exakt diesen Positionen übernommen. Die im Kind existierenden Lücken, werden mit den fehlenden Elementen in genau der Reihenfolge bestückt, wie sie im zweiten Elternteil auftreten.[Sysw91]

Bei näherer Betrachtung und Berücksichtigung des verwendeten Greedy-Platzierungs-Algorithmus wurde deutlich, dass der Einfluss der Erhaltung von Nachbarschaften eine starke Abhängigkeit von deren Position in der Einfügereihenfolge hat. Die Erhaltung einer Nachbarschaft im hinteren Verlauf der Reihe verliert ihren Wert, wenn im vorderen Teil viele Vertauschungen stattfinden. Denn für

eine Platzierung der Nachbarschaft bilden die früh eingesetzten Teile die geometrische Grundlage. Im Gegenzug ist eine sehr gute Lösung stark abhängig vom vorderen Teil ihrer Platzierungsreihenfolge.

KeepStartingSequence: Für diese Arbeit wurde eine Rekombination entworfen, bei der eine zufällige Anzahl an Elementen zu Beginn der Reihenfolge vom ersten Elternteil übernommen wird. Dies soll ermöglichen, dass bewährte Startsequenzen bei der Vererbung erhalten bleiben. Es wird über den Parameter des definierten Anteils an zu übernehmenden Elementen eine zufällige Position gewählt bis zu der alle Elemente ab der Startposition übernommen werden. Falls der Anteil noch nicht erreicht ist, werden weitere Positionen im Verfahren wie beim Position Crossover vererbt.

	Elternteil 1	×	Elternteil 2	Nachkomme
Single Crossover	{**5,2,7**, 6, 1, 3, 4}		{7, 6, 5, 3, 2, 1, 4}	{**5,2,7**, 6, 3, 1, 4}
Position Crossover	{5, **2**, 7, **6**, 1, **3,4**}		{7, 6, 5, 3, 2, 1, 4}	{7, **2**, 5, **6**, 1, **3,4**}
KeepStartingSequence	{**5,2,7,6**, 1, 3, 4}		{7, 6, 5, 3, 2, 1, 4}	{**5,2,7,6**, 3, 1, 4}

Die Nummern beschreiben in diesem Beispiel die Sortierung der Geometrien

Abbildung 13. Beispielausprägungen Rekombinationen

Die Parametrierung der Rekombinationsoperationen ist dabei Bestandteil der Voruntersuchung zur Validierung und wird in über Versuchsreihen erörtert.

4.4.8 Mutation In Anlehnung an biologische Evolutionsprozesse werden die Elemente einer Population einzeln mutiert um die Möglichkeit zu nutzen durch kleine Änderungen der Chromosomausprägungen eine bestehende Lösung stückweise zu verbessern. Folgende Mutationen werden im Rahmen der Arbeit umgesetzt und deren Einsatz geprüft:

Vertauschung von Nachbarn: Beim sogenannten Neighbour Swap wird an einer zufälligen Stelle der Reihenfolge innerhalb eines Individuums ein Positionstausch zweier Nachbarn durchgeführt. Es ist darauf zu achten, dass die Nachbarn sich von Geometrie und Ausprägung unterscheiden.

Startposition: Da die initiale geometrische Position eines Teils Einfluss auf seine erfolgreiche Platzierung hat, wird mit dieser Mutation die Startposition verändert. Wenn diese Mutation in Verbindung mit einem Bottom-Left-Algorithmus kombiniert wird, kann die Mutation auf eine Achse beschränkt werden, da eine Verschiebung in der anderen Achse keine Vorteile mit sich bringt.

Umdrehen der Reihenfolge: Eine weitere Operation, ist die stückweise Umkehrung (ReverseOrder) der Sortierung. Es wird ein zufällig bestimmter Bereich der Reihenfolge ausgewählt und die Elemente in umgekehrter Reihenfolge wieder eingefügt.

Zufälliges Einfügen: Bei der sogenannten RandomInsertion wird ein Element der Liste entnommen und an einer anderen Stelle neu eingefügt. Wie bei einer logischen Kette wird dabei die Restkette jeweils um ein Element verschoben.

Zufällige Rotation: Da die Orientierung der Geometrien einen großen Einfluss auf deren gute Platzierbarkeit hat, wird bei der RandomRotation die Ausgangsdrehung des Elements verändert. Hierbei kann vorab konfiguriert werden in welcher Schrittweite dies möglich ist um ungültige Lösungen zu vermeiden.

Zufälliger Tausch: Zwei zufällig ausgewählte Elemente der kompletten Liste werden miteinander getauscht. Bei diesem RandomSwap sollte darauf geachtet werden, dass die beiden Elemente sich von Geometrie und Ausprägung unterscheiden.

Durcheinanderwürfeln: Durch die geordneten und nachbarschaftserhaltenden Mutationen sind manche tiefgreifende Umsortierungen mit besserer Gesamtlösung schwierig zu erreichen. Deshalb wird durch die Mutation Shuffle ein Teil der Liste zufällig vermischt in der Hoffnung, eine passendere Sortierung erhalten zu haben.

Rotieren der Reihenfolge: Bei der Mutation RotateRight wird ein Teil einer Einfügereihenfolge um eine Stelle verschoben, wobei das dabei „herausgefallene“ Element in die entstehende Lücke wieder eingefügt wird. Hierbei ist ein Erhalten der Nachbarschaftsbeziehungen in großen Teilen gegeben.

Neuplatzierung des ersten Elements: Das erste Element einer Einfügereihenfolge hat großen Einfluss auf die Platzierung durch einen Greedy-Algorithmus. Bei dieser Mutation wird daher das erste Element an eine zufällige Position der Liste verschoben.

Neuplatzierung des „schlechtesten" Elements: Die Bewertung einer Lösung findet oft durch die Berechnung der Gesamthöhe statt. Nun kann es vorkommen, dass ein komplexes, großes oder schwer zu platzierendes Teil so positioniert wird, dass es mit einer seiner Eckpunkte den höchsten Punkt definiert und damit die Bewertung definiert. Es muss dabei nicht zwangsläufig das letztplatzierte Element sein. Dieses Element wird an zufälliger Position in die Liste eingefügt um bei einer erneuten Platzierung gegebenenfalls an einer günstigeren Position zu einer besseren Gesamtlösung beizutragen.

Ausgangsausprägung →	nach der Mutation
ReverseOrder $\{1,2,3,\mathbf{4,5,6},7\}$	$\{1,2,3,\mathbf{6,5,4},7\}$
NeighbourSwap $\{1,2,3,\mathbf{4,5},6,7\}$	$\{1,2,3,\mathbf{5,4},6,7\}$
RandomInsertion $\{1,\mathbf{2},3,4,5,6,7\}$	$\{1,3,4,5,\mathbf{2},6,7\}$
RandomSwap $\{\mathbf{1},2,\mathbf{3},4,5,6,7\}$	$\{\mathbf{3},2,\mathbf{1},4,5,6,7\}$
Shuffle $\{1,\mathbf{2,3,4,5,6},7\}$	$\{1,\mathbf{4,6,5,3,2},7\}$
RotateRight $\{1,2,\mathbf{3,4,5,6,7}\}$	$\{1,2,\mathbf{7,3,4,5,6}\}$
FirstElement $\{\mathbf{1},2,3,4,5,6,7\}$	$\{2,3,4,5,\mathbf{1},6,7\}$
WorstElement $\{1,2,3,4,5,\mathbf{6},7\}$	$\{1,2,3,\mathbf{6},4,5,7\}$

Die Nummern beschreiben in diesem Beispiel die Sortierung der Geometrien

Abbildung 14. Beispielausprägungen (Reihenfolgen-)Mutationen

Durch die große Zahl an Parameterausprägungen und möglichen Mutationen sind umfangreiche Untersuchungen nötig, um korrekte Parameter zu finden. Dabei handelt es sich sowohl um die Konfiguration der Mutationen als auch die Untersuchung welche Mutationen Verwendung finden sollen und mit welcher Wahrscheinlichkeitsverteilung diese ausgewählt werden.

Laut Pohlheim [Pohl13] wird die Mutationsrate meist auf (1/Anzahl der Variablen) gesetzt. Dies soll für die Parameterfindung in dieser Arbeit als Einstiegspunkt dienen.

4.4.9 Wiedereinfügen Um Platz für die neuen möglichen Lösungen zu schaffen, müssen Individuen aus der Population ausgeschlossen

werden. Für eine neue Generation findet eine direkte Auswahl statt. Dabei gibt es mehrere Kennzahlen:

Wiedereinfügerate: Es wird beschrieben, wie viele Individuen einer Generation beim Wechsel in die Folgegeneration ersetzt werden. Eine Rate von 1 bedeutet, dass alle Individuen ersetzt werden. Da dies die Gefahr birgt, dass eine bereits berechnete beste Lösung durch eine schlechtere ersetzt wird, wählen wir eine Rate von knapp unter 1. Sodass mindestens die beste Lösung erhalten bleibt. Dies ermöglicht es, neuen Individuen in die Population einzutreten ohne die beste Lösung zu verlieren.

Generationslücke: Diese Rate gibt an wie viele Nachkommen pro Individuum erzeugt werden. Da in diesem Fall mehrere Mutationen angewendet werden und ein Teil einer Generation durch eine Rekombination gekreuzt wird, tritt eine Generationslücke von 6 auf.

Einfügevorschrift: Aufgrund der Tatsache, dass mehr Individuen generiert werden, als in die nächste Generation übernommen werden können, muss eine Selektion stattfinden. Anhand der Wiedereinfügerate werden die schlechtesten Elemente der vergangenen Generation entfernt. Dann werden die besten Lösungen der aktuelle erzeugten Nachkommen und Mutationen eingefügt, bis die gewünschte Größe der Population wieder erreicht ist. Dies ist die sogenannte Truncation-Selection. [Pohl13] Der sogenannte Selektionsdruck beträgt daher 1, da stets nur die besten Individuen in die nächste Generation übernommen werden.

Die Lebensdauer der Individuen verhält sich dabei in Abhängigkeit der beiden Parameter und wird in Generationen pro Individuum angegeben. Im behandelten Fall hätte ein Individuum daher eine durchschnittliche Lebensdauer von circa 0,2 Generationen.

$$\varnothing Lebensdauer = \frac{1}{\text{Wiedereinfügerate} \times \text{Generationslücke}} \tag{7}$$

4.4.10 Initialisierung der Population Ein evolutionärer Algorithmus braucht zu seinem Beginn mindestens eine bestehende Lösung für den Start des Optimierungsprozesses. Dazu werden so

viele Individuen aus der gegebenen Menge und Reihenfolge von Teilen permutiert, bis die Startgeneration voll besetzt ist.

Um für häufig auftretende Charakteristika der Eingabemenge möglichst günstige Startbedingungen zu erhalten, werden mehrere Individuen nach gegeben Kriterien sortiert:

- Länge des Objekts
- exakte Fläche des Objekts
- Füllgrad bzgl. des einhüllenden Rechtecks
- Höhe des Objekts
- Fläche des einhüllenden Rechtecks

4.4.11 Abbruchkriterium Der evolutionäre Algorithmus ist durch den iterativen Charakter so konzipiert, dass er sich beliebig oft hintereinander ausführen lässt und bestrebt ist, die bestmögliche Lösung zu finden. Da dies theoretisch ein unendlich andauernder Prozess sein kann, ist es nötig zu definieren, bei welchem Ereignis das beste bisher bekannte Individuum als Lösung herausgegeben wird. Es ist dabei zu beachten, dass die relative Verbesserung einer Lösung bei zunehmender Simulationsdauer durchschnittlich stark abnimmt und somit eine grobe Schätzung des weiteren Verlaufs möglich ist. Um das Verhältnis von Rechenaufwand zu Lösungsgüte zu kontrollieren gibt es folgende Abbruchkriterien:

Direkte Kriterien: Die Zählstände von insgesamt berechneten Individuen oder Generationen finden als direkte Kriterien Verwendung. Auch möglich ist die einfache Beschränkung der Simulationszeit auf eine gegebene Dauer. Je nach Zielsetzung der Anwendung kann auch die Differenz zu einem bekannten Optimum oder die vorgegebene Zielgröße der Wertefunktion angegeben werden.

Abgeleitete Kriterien: Neben den direkten Kriterien ist es möglich, die Werte innerhalb einer Generation zu vergleichen und daraus abzuleiten, wie vielversprechend die Generation in Bezug auf weitere Verbesserungen ist. Dies kann über die Standardabweichung, einen laufenden Mittelwert, die Differenz der Extreme oder Kappa (Gleichartigkeit der Ausprägungen verschiedener Individuen) geschehen. Es gilt speziell zu beachten, dass nicht bei allen Kriterien das Erreichen des Abbruchpunktes garantiert ist.

Die genannten Kriterien schließen sich nicht gegenseitig aus, sondern können in der Praxis durch verschiedene logische Operanden kombiniert werden.

5 Implementierung

Nach dem Entwurf des zu verwendenden Algorithmus wurde dieser und das zugehörige Framework implementiert, um die Leistungsfähigkeit und Verwendbarkeit in den Anwendungsfällen zu prüfen.

5.1 Umgebung

Als Laufzeitumgebung wurde die Java Virtual Machine ausgewählt. Aufgrund der Plattformunabhängigkeit in späteren Schritten ist es leichter, Laufzeitversuche auf Clustern, Servern, regulären Computern und eingebetteten Systemen durchzuführen. Zudem bietet die Umgebung eine einfache Integration von Bibliotheken und Testmechanismen wie Unittests. Die Programmiersprache ist in diesem Falle Java. Trotz des bereits erschienenen Java Runtime Environment 8 wurde aus Kompatibilitätsgründen das Java Runtime Environment 7 verwendet.

5.2 Aufbau

Das Programm selbst ist als eigenständige Software konzipiert und bedient sich keiner zusätzlichen Software. Es gibt keine grafische Benutzeroberfläche, sodass die Simulation direkt über die Konsole gestartet wird. Dabei wird als Parameter der Dateiname der auszuführenden Simulationskonfiguration übergeben.

Allgemein ist die Simulation nach den Grundprinzipien der objektorientierten Programmierung implementiert und es wurde im Vorhinein ein Grobgerüst unter Verwendung von Entwurfsmustern aufgesetzt. Dieses Vorgehen hat über die Zeit der Erstellung einen hohen Grad an Flexibilität und Austauschbarkeit garantiert, dank diesem sich weiterhin neue Komponenten hinzufügen lassen.

Für Ein- und Ausgabedaten wurde das Format SVG (Scalable Vector Graphics) gewählt um eine einfache Bedienung und Anwendungsnähe zu ermöglichen.

5.2.1 Bibliotheken Im Rahmen der Implementierung wurden verfügbare Bibliotheken verwendet um Teilfunktionalitäten direkt nutzen zu können.

Standard: Das in der Java Plattform SE verfügbare Paket *java.awt.geom* wurde für Geometrieoperationen verwendet und diente als Grundlage für weitere Implementierungen. Darüber hinaus wurden vor allem im Bereich des Dateizugriffs und Listenverwaltung bestehende Pakete verwendet.

SVG Handling: Für den Einleseprozess der Eingabegeometrien aus dem SVG-Format wurde das Java SVG Toolkit Batik 1.8 von Apache verwendet. Dieses stellt ein stabiles Einlesen von Vektordaten mit verschiedenen Spezifikationen und Darstellungsvarianten sicher. Als hierarchische Struktur lässt sich einfach auf die Pfade und Pfadstücke zugreifen und diese in die zur Berechnung benötigte Modellierung überführen.

5.3 Algorithmus

Zur besseren Übersichtlichkeit und einfachen Entwicklung von alternativen Ansätzen wurden zu allen verwendeten Berechnungsbausteinen, je nach Komplexität, Interfaces und abstrakte Klassen angelegt. Dies ermöglicht eine klare Trennung der verschiedenen Algorithmenbestandteile, vereinfacht die Durchführung von Tests und ermöglicht es, Programmbausteine schnell auszutauschen. Ein Beispiel für eine solche Implementierung ist der evolutionäre Algorithmus, dessen Ablauf in der abstrakten Klasse komplett definiert ist und dessen Inhalte der einzelnen Algorithmenschritte durch die implementierende Klasse definiert werden.

5.4 Modell

Bei der Umsetzung des Modells mussten einige laufzeitrelevanten Umsetzungen geprüft werden, um spätere Performancenachteile bei der Simulation zu verhindern. Dazu wurde vor allem der Zugriff auf die Geometriedaten und deren Repräsentation und Modifikation überarbeitet. Besonders anfällig war der Umgang mit den vielen Instanzen einer Ursprungsgeometrie, welche aufgrund der Mutationsoperatoren in vielen Variationen vorliegen kann. Eine Möglichkeit wäre das Einführen einer zusätzlichen Information über die angewendeten Transformationen als eigenes Objekt, das auf das Ursprungsobjekt

referenziert. Die Alternative wäre, eine komplette Kopie der Geometrie und aller seiner verknüpften Objekte zu erstellen und diese direkt zu modifizieren.

TransformationsObjekt	*DeepCopy*
+ Direkte Referenz auf das Originalobjekt	+ Einfacheres Handling
+ Keine Kopieroperationen	+ Einfachere Parallelisierung
− Viele Rechenoperationen	− Redundante Informationen
	− Erhöhter Speicherbedarf
	− Rekombination schwieriger

Nach einer Laufzeitanalyse wurde die DeepCopy Lösung umgesetzt, da der erhöhte Speicherbedarf wesentlich weniger Einfluss auf die Laufzeit hat, als der erhöhte Rechenaufwand für Verschiebungs- und Kollisionsabfragen.

5.5 Applikation

Aufgrund der Komplexität der zugrundeliegenden Problemstellung wurde keine grafische Benutzerschnittstelle implementiert. Die Bedienung des Systems findet über den einfachen Konsolenaufruf statt und wird über vorgegebene Struktur und Speicherorte von Ein- und Ausgabedaten ausgeführt. Konfigurationseinstellungen werden dabei auch über eine Parameterdatei eingelesen.

5.6 Optimierungen

Während der Implementierung wurden über Profiler Funktionen identifiziert, die einen hohen Anteil an der gesamten Berechnung einnehmen. Entweder durch eine lange Einzellaufzeit oder durch den vielfachen Aufruf. Um die Simulationszeit bzw. die Iterationszeit zu verkürzen wurden unter anderem folgende Maßnahmen umgesetzt:

5.6.1 Redundante Berechnungen eliminieren Bei der Laufzeitanalyse wurde deutlich, dass durch die Kollisionsabfrage sehr oft die einhüllenden Geometrien (BoundingBoxes) von Elementen angefragt wurden. Da diese zumeist aus den bestehenden Eckpunkten des Objekts berechnet werden müssen, ist dies ein großer Rechenaufwand. Im Rahmen eines Durchlaufs des Platzierungsalgorithmus

kommt es vielfach vor, dass die Einhüllende einer Geometrie angefordert wird. Da sich das Objekt, z.B. wenn es als erstes platziert wurde, dazwischen oft nicht bewegt, hat die Berechnung jeweils das gleiche Ergebnis. Um Aufwand einzusparen wird die Einhüllende dauerhaft im Objekt gespeichert und alle modifizierenden Funktionen wurden so erweitert, dass sie die Einhüllende nach einer Modifikation aktualisieren. Dies führte zu einer deutlichen Beschleunigung des Simulationsdurchlaufs.

Eine ähnliche Situation finden wir bei der Berechnung der Zielfunktion vor. Da diese pro Individuum stets gleichbleibend ist, sorgt eine Zwischenspeicherung für verbessere Laufzeiten bei Sortieroperationen. Nach einer Mutation oder Rekombination wird einmalig eine neue Bewertung durchgeführt und abgespeichert.

5.6.2 Überspringen von Leerraum Je nach gewählter Implementierung des Greedy-Algorithmus kann es vorkommen, dass Objekte in diskreten Schritten zur endgültigen Platzierung geführt werden. Die Bewegung von der Startposition in die Nähe der bereits platzieren Objekte findet daher im Leerraum statt. Da diese Bewegung unnötige Rechenzeit in Anspruch nimmt, wurden verschiedene konzeptionelle und implementatorische Maßnahmen ergriffen, um den Leerraum zu umgehen. Über die gegebene räumliche Repräsentation als Uniform Grid ist es einfach festzustellen, welche Felder komplett leer sind und einfach übersprungen werden können. Zudem können nach dem bewährten Verfahren der binären Suche Objekte effizient mit hoher Genauigkeit platziert werden, ohne, dass große Abhängigkeit von der gegebenen Auflösung bzw. Diskretisierung besteht.

5.7 Parametrierung

Zur Simulation ist es nötig, eine Auswahl der zur Verfügung stehenden Komponenten (Mutationen, Rekombinationen) zu treffen und zudem die damit verbundenen Parameter passend zu setzen. Ziel ist es, dabei mit einem vertretbaren Zeitaufwand gute Ergebnisse berechnen zu können. Es werden als Ausgangswerte bekannte Werte aus der Literatur verwendet. Aus dem Einführungswerk für Evolutionäre Algorithmen von Pohlheim [Pohl13] sind folgende Werte als Startwerte entnommen:

- Wiedereinfügerate: 0,90-0,99
- Generationslücke: 3-10
- Rekombinationsrate: 0,7
- Selektionsdruck: 1
- Mutationsrate: 1/Variablenanzahl

Für eine ausreichende Diversität an Startpopulationen wurde eine Populationsgröße von 100 Individuen als Ausgangspunkt gewählt. Zur Beobachtung des Verlaufs wurde als Abbruchkriterium eine maximale Generationsanzahl von 350 Generationen gewählt.

Experimentell wurde untersucht, wie die Einflüsse von Änderungen dieser Startwerte auf die Ergebnisse der Berechnung sind und diese durch Herleitungen nachvollzogen. Nach einigen Simulationsläufen haben sich folgende Parametereinstellungen bewährt:

Wiedereinfügerate Die Einfügerate wurde auf 0,95 festgesetzt, da bei näherer Untersuchung eine höhere Rate dazu geführt hat, dass sich ausgehend von der Startpopulation zu schnell nur die Nachfahren eines Individuums durchsetzen und somit sehr schnell ein Diversitätsverlust auftritt. Bei niedrigeren Einfügeraten verbleiben viele Lösungen in Population, welche schon mutiert und kombiniert wurden und blockieren somit den Platz für neue Individuen. Diese können teilweise zwar eine schlechtere Bewertung haben, bieten aber eher die Möglichkeit, damit lokale Minima zu verlassen.

Generationslücke Die Generationslücke gibt an wie viele Nachfahren oder Mutationen pro Individuum erzeugt werden. Dies ist vor allem abhängig von der Art der Berechnung und Komponentenkonfiguration des Algorithmus. Durch Rekombination und Mutation werden für jede Generation sechs neue Lösungen aus einem Individuum generiert.

Rekombinationsrate: Der verwendete Wert für die Rekombinationsrate aus der Literatur konnte auf die im Rahmen dieser Arbeit umgesetzte Rekombinationsvariante KeepStartingSequence nicht übertragen werden. Ein zu großer Wert führt zu einer zu großen Übereinstimmung zwischen erstem Elternteil und Kind, sodass in Verbindung mit der Auswahl der besten Individuen zur Rekombination sehr schnell nur noch sehr ähnliche Lösungen generiert werden. Es wurde nach mehreren Durchgängen herausgefunden, dass eine Rate von 0,33

eine ausreichende Stärke der Elternausprägung vererbt wird und gleichzeitig eine große Varianz möglich bleibt.

Selektionsdruck: Der Selektionsdruck wurde beim Maximalwert von 1 beibehalten, um sicherzustellen, dass neu generierte Lösungen, welche besser als die bisher beste Lösung sind, auch sicher in die Folgegeneration übernommen werden.

Mutationsrate: Die bewährte Rate wurde beibehalten, um zu verhindern, dass ein Individuum durch zu viele Mutationen sich zu stark verändert und so der evolutionäre Effekt geschwächt wird.

6 Validierung

Nach Entwurf und Implementierung gilt es die erarbeiteten Inhalte auf ihre Tauglichkeit zu prüfen und in den Vergleich zu bestehenden Systemen Dritter zu stellen. Es wurden im Rahmen der Arbeit Validierungen auf verschiedenen Ebenen und Anwendungsfällen durchgeführt, welche folgend im Detail erläutert werden.

6.1 Vorgehen

Um die Korrektheit von selbst implementierten Grundfunktionen zu prüfen, wurden entwicklungsbegleitend JUnit-Tests durchgeführt. Speziell bei den geometrischen Operationen und bewertenden Programmbausteinen wurden umfangreiche Tests zur Sicherung der Korrektheit implementiert.

6.2 Voraussetzungen

Um eine Lösung mit Anderen vergleichen zu können, muss sichergestellt sein, dass es sich um eine korrekte Lösung handelt. Hierzu wurden Konsistenzprüfungen implementiert, die Ergebnisse und Zwischenergebnisse prüfen und die Simulation gegebenenfalls unterbrechen. Durch die Implementierung des Algorithmus ist eine hinsichtlich Anzahl, Überlappung und Geometrietreue konsistente Lösung gegeben, wenn ein Ergebnis zurückgegeben wird. In den gestellten Platzierungsaufgaben muss der Algorithmus stets eine gültige Lösung generieren, die bewertet werden kann.

6.3 Zielkriterien

Aufgrund des vielfältigen Betätigungsfeldes besteht kein allgemeingültiges und etabliertes Zielkriterium. Im Bereich des Lederzuschnitts [HeLe95] handelt es sich um eine Wertefunktion der Güte der verwendeten Fläche. Bei den klassischen Platzierungsbenchmarks mit Anlehnung an die Großindustrie [ShKa13] sind die einfachen Längen des benötigten Rohmaterials ausreichend.

Um Lösungen in Relation setzen zu können, müssen daher Kriterien ausgewählt werden, nach denen verglichen werden kann. Hierzu sollen die Anwendungsfälle berücksichtigt werden, bei denen es um

die Nützlichkeit der gegebenen Lösung und der weiteren Verwendung des Rohmaterials geht.

6.4 Bewertungsfunktionen Materialausnutzung

Primäres Ziel aller Anwender ist die Einsparung von Kosten. Dies stimmt in fast allen Fällen direkt mit der Fläche der nicht mehr verwendbarenIm Sinne des Trennprozesses und nicht im Weiterverkauf des Wertstoffs Teile des Rohmaterials überein. Nur bei großindustriellen Prozessen wird Materialverlust in Kauf genommen, wenn dadurch andere Kosteneinsparungen durch kürzere Laufzeiten oder höheren Automatisierungsgrad erreicht werden können. Da sich diese Arbeit aber mit Kleinserien und Prototypenbau beschäftigt, wird dieser Anwendungsfall nicht näher betrachtet. Das oberste Zielkriterium, das gemessen werden muss, ist daher die genutzte Fläche bzw. die noch nutzbare Restfläche.

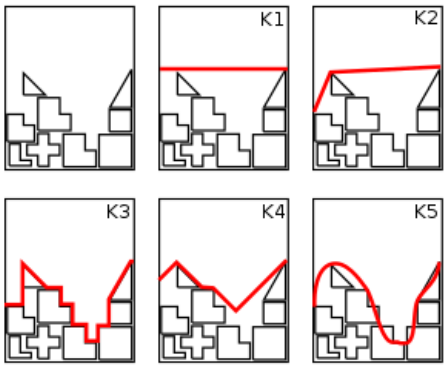

Abbildung 15. Visualisierung der Bewertungsfunktionen

6.4.1 Höhe des angeschnittenen Materials In Industriezweigen, in denen es um prozentual große Einsparungen bei der Verwendung von Endlosmaterial oder Rollenmaterial geht, ist das klassische Bewertungskriterium die Fläche des benötigten Materials. Die Breite ist durch das Rohmaterial bereits definiert. Dies ist daher auch die Bewertungsfunktionen des Strip-Packagings, welches in der Forschung

viel für Benchmarks verwendet wird. Für Platzierungen mit geringen Stückzahlen, welche beispielsweise nicht die ganze Breite des Rohmaterials ausnutzen können, ist diese Berechnung wenig aussagekräftig. Eine Visualisierung der Funktion ist in Abbildung 15 unter K1 zu sehen.

$$Bewertung = \frac{\text{Fläche des achsensymmetrischen, umschließenden Rechtecks}}{\text{Gesamtfläche des Rohmaterials}} \tag{8}$$

6.4.2 Konvexe Hülle Bei nicht annähernd geradlinig begrenzten Platzierungen sinnvoll die genutzten Fläche zu bewerten, ist aufwändig. Um möglichst wenig noch nutzbares Material dem Verschnitt zuzurechnen, lässt sich über die konvexe Hülle eine geeignete Näherung als durch die reine Höhe berechnen. Die konvexe Hülle eignet sich auch für irregulär begrenztes Rohmaterial. Dennoch kann sie in vielen Fällen zu großen Bewertungsabweichungen führen, bei der nutzbares Material unterschlagen wird oder gegenteilig nicht nutzbares Material am Rand eingerechnet wird. Eine Visualisierung der Bewertungsfunktionen ist in Abbildung 15 unter K2 zu sehen.

6.4.3 Projektion Die Ansätze der rechteckigen Höhenmessung liefert in allen Fällen eine sichere Abschätzung, da das wieder zu verwertende Restmaterial rechteckig definiert ist und somit keine Hindernisse durch vorangegangen platzierte Teile bestehen. Leider führt dies bei speziellen Geometrien zu starken Abweichungen, da oft großflächige Stücke dem Verschnitt zugeordnet werden, obwohl diese in einem weiteren Prozess nutzbar wären. Um eine genauere Abschätzung zu erhalten, ist es nötig, die Verschnittfläche durch einen Pfad anstatt durch eine waagerechte Linie zu begrenzen . Da vermieden werden soll, dass zu viel Material als Wertmaterial erkannt wird, ist es wichtig, Hinterschnitte zu beachten und tiefe Einschnitte auf ihre Verwendbarkeit zu beurteilen. Eine Visualisierung als einfache Projektion ist als durchgezogene Linie in Abbildung 15 unter K3 zu sehen.

6.4.4 Winkelgrenzen Um das Problem tiefer Einschnitte bei der Projektion und dem Stufenmodell zu umgehen, kann der maximale

Winkel einer begrenzenden Linie beschränkt werden: Damit wird eine Glättung erreicht. In Abbildung 15 unter K4 ist zu erkennen, dass große Lücken dennoch als noch zu verwertendes Material gewertet werden.

6.4.5 Bezierkurven Statt dem Versuch, die Fläche möglichst exakt zu begrenzen ist der Ansatz über Bezierkurven eine Heuristik, die den Verbrauch nähert. Die Aussagekraft der Bewertung beruht auf der Annahme, dass stets ähnlich große Bereiche hinzugerechnet wie abgezogen werden. (Siehe Abbildung 15 unter K5)

6.4.6 Weitere Variationen Um Ungenauigkeiten der verschiedenen Bewertungsfunktionen auszugleichen und Sonderfälle sicher abdecken zu können, lassen sich die Funktionen auch kombinieren. Eine gemeinsame Bewertung kann dann aus den Einzelbewertungen berechnet werden. Alternativ ist es möglich, die Bewertung über mehrere Ebenen durchzuführen, sodass bei Gleichheit von Bewertungen eine Andere für den direkten Vergleich und die relative Ordnung hinzugezogen werden kann.

Zudem ist es möglich, die Bewertung über eine nichtlineare Berechnung zu erzeugen, um bestimmte Bereiche besser unterscheiden zu können.

6.4.7 Auswahl Die Vor- und Nachteile der verschiedenen Verfahren beziehen sich in fast allen Fällen auf Sonderfälle oder spezielle Einsatzgebiete. Da im Rahmen der Validierung vor allem der Vergleich mit anderen Verfahren und bestehenden Benchmarks gesucht wird, ist die einfache Berechnung der Höhe die geeignete Funktion. Bei genauerer Betrachtung ist dieses Kriterium auch für die in Paragraph 6.4.1 genannten Sonderfälle geeignet, wenn die tatsächlichen Anwendungsfälle berücksichtigt werden. Die Näherung der Trennkante als waagrechte Linie ist doch geeignet (siehe Abb. 16), wenn mit mehreren Eingabemengen auf dem selben Material mehrfach der Platzierungsalgorithmus ausführt wird.

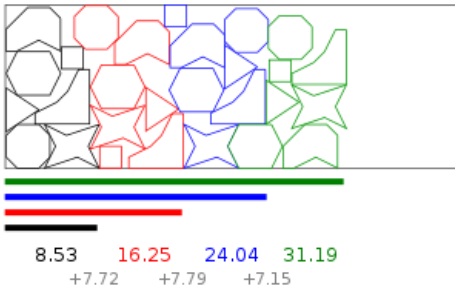

Abbildung 16. Anwendung des Kriteriums bei der mehrfachen Platzierung

6.5 Testdaten

Die zur Durchführung der Validierung benötigten Datensätze sollen Anwendungsfällen möglichst genau entsprechen sowie eine gute Vergleichbarkeit zu anderen Verfahren bieten. Im Folgenden werden verfügbare Datenquellen genauer erläutert und die möglichen Testdaten beschrieben.

6.5.1 Benchmarks

Blaz: Ein weit verbreiteter und vielfach verwendeter Benchmark für Tests von Platzierungsverfahren im Strip-Packaging ist die von Oliveira und Ferreira[OlGF00] vorgestellte Sammlung fiktiver Geometrien. Es handelt sich um insgesamt sieben Konturen, welche in zwei Konfigurationen Verwendung finden (siehe Abb.17). Die Umrisse sind eindeutig konstruiert und nur für Testläufe gedacht. Aufgrund des stets vierfachen Auftretens der einzelnen Geometrien, eignet sich der Benchmark, um in dieser Arbeit als Variation verwendet zu werden. Für mehrfach hintereinander angewendete Platzierungsvorgänge kann so untersucht werden, wie gut solche Platzierungsprobleme stückweise gelöst werden können.

Titel des Benchmarks	blaz1	blaz2
Anzahl verschiedener Teile	7	4
Anzahl Teile insgesamt	28	20
∅ Anzahl Eckpunkte	6,29	7,5
Erlaubte Rotationswinkel (absolut)	0°;180°	0°;180°
Rohmaterialbreite	15	15

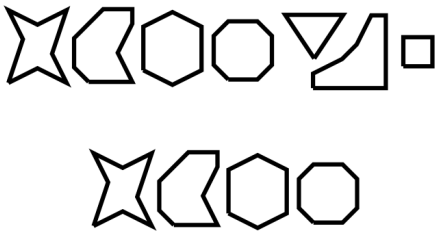

Abbildung 17. Alle Teile des Benchmarks Blaz1 (oben), Blaz2 (unten)

Jakobs: Als Vergleichsgrundlage in vielen wissenschaftlichen Veröffentlichungen werden die Datensätze aus der Arbeit von Jakobs[Jako96] herangezogen. Auch diese sind fiktive Geometrien zur Validierung von Strip-Packaging Algorithmen. Es handelt sich dabei um je 25 Einzelkonturen, welche sich teilweise nur durch verschiedene Skalierungen unterscheiden (siehe Abb.18/19). Durch die hohe Diversität der Teile gestaltet sich die Platzierung als große Herausforderung und wird daher als guter Benchmark betrachtet.

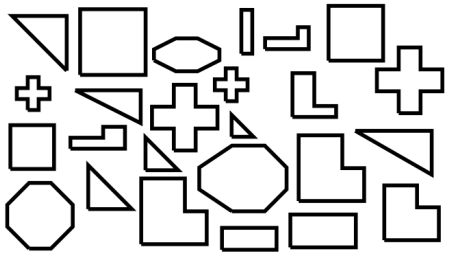

Abbildung 18. Alle Teile des Benchmarks jakobs1

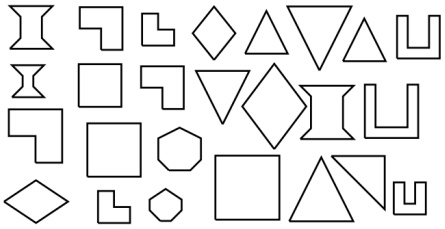

Abbildung 19. Alle Teile des Benchmarks jakobs2

Titel des Benchmarks	jakobs1	jakobs2
Anzahl verschiedener Teile	25	25
Anzahl Teile insgesamt	25	25
∅ Anzahl Eckpunkte	5,6	5,36
Erlaubte Rotationswinkel (absolut)	0°;90°;180°;270°	0°;90°;180°;270°
Rohmaterialbreite	40	70

Swim: Ein Datensatz, der realen Fertigungsdaten entnommen ist, wurde von Oliveira und Ferreira[OlGF00] vorgestellt und bildet die Textilstücke einer Schwimmbekleidung ab. Auch dieser Datensatz findet für den Vergleich von Strip-Packaging Algorithmen Verwendung. Die Rohdaten sind über Vektorangaben definiert, haben aber wesentlich größere Abmaße als die Geometrien der anderen Benchmarks.

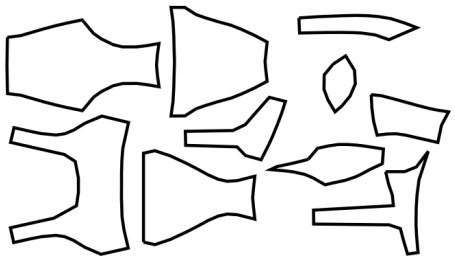

Abbildung 20. Teile des Benchmarks swim

Titel des Benchmarks	swim
Anzahl verschiedener Teile	10
Anzahl Teile insgesamt	48
∅ Anzahl Eckpunkte	21,9
Erlaubte Rotationswinkel (absolut)	0°;180°
Rohmaterialbreite	5752

6.5.2 Architekten Im klassischen Modellbau für den Gebäudebau ist der Zuschnitt von maßstabsgetreuen Kleinteilen wesentlicher Bestandteil des Berufsbildes. Durch die fortschreitend computerunterstützte Planung existieren die Baupläne heute stets auch digital. Daher liegt es nahe, auch den bisher händischen Zuschnitt durch Laserschnitt zu ersetzen. Im Fachgebiet Building Lifecycle Management am Karlsruher Institut für Technologie wird dieser Prozess für studentische Arbeiten bereits durchgeführt. Zur Verwendung im Rahmen der Validierungsaktivitäten wurde ein Datensatz eines Gebäudemodells bereitgestellt. Es handelt sich dabei um großflächige Gebäudewände (vgl. Abb. 21).

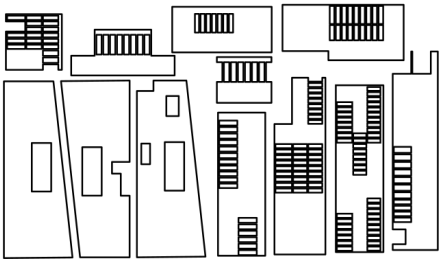

Abbildung 21. Auswahl von Geometrien im Umfeld des architektonischen Modellbaus

6.5.3 Prototypenbau im technischen Kontext Für den Bau von Prototypen im Bereich von Aerodynamik- und Fahrzeugstrukturbauteilen nutzt die Hochschulgruppe KA-RaceIng e.V. einen Lasercutter für den Zuschnitt von Formen, Versteifungen und Schaumeinsätzen. Die zur Verfügung gestellten Daten bieten interessante Charakteristika über die verschiedenen verwendeten Materialien. Die

gezeigten Holzteile (siehe Abb. 22) sind sehr lang und haben fast keine geradlinigen Kanten.

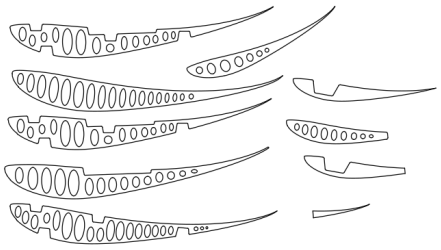

Abbildung 22. Auswahl von Geometrien im Umfeld des technischen Prototypenbaus

6.5.4 Startup Im Bereich des Prototypenbaus und der Kleinserienfertigung von Funktionsmodellen wurden für die Validierung des Algorithmus von der Firma LehrWerk verschiedene Rohdaten zur Verfügung gestellt. Neben den einfachen Geometrien wurden auch händisch angeordnete Schnittaufträge sowie Geometrien von üblichem angeschnittenem Rohmaterial bereitgestellt(vgl. Abb. 23). Durch die Produktform ist eine deutliche Charakteristik der Eingabedaten zu erkennen. Viele der Teile sind annähernd kreisförmig. Auffällig häufig sind die Teile mit großen Ausschnitten im Inneren versehen, welche für kleinere Teile als verwertbares Rohmaterial verwendet werden können. In der Praxis ist dies jedoch nur relevant, wenn die Verwertung im gleichen Prozessschritt stattfindet.

6.5.5 Prototypenbau im wissenschaftlichen Kontext Im Rahmen von Lehre und Forschung wird am Institut für Produktentwicklung am Karlsruher Institut für Technologie ein Lasersystem eingesetzt. Es werden dabei verschiedenste Anwendungsfälle auf einer Maschine durchgeführt. Die Bandbreite reicht von technischen Prototypen über Getriebemodelle bis hin zu Gehäusen und Beschriftungstafeln (vgl. Abb. 24).

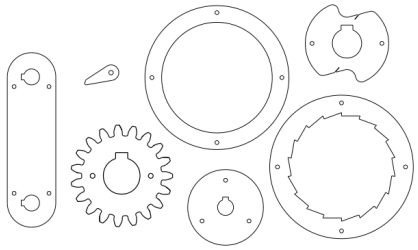

Abbildung 23. Auswahl von Geometrien im Umfeld des funktionalen Modellbaus

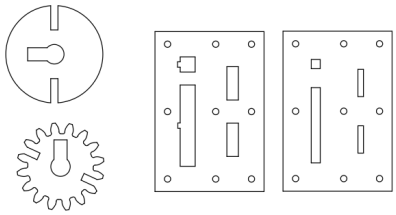

Abbildung 24. Auswahl von Geometrien im wissenschaftlichen Umfeld

6.5.6 Sonderfälle Im Rahmen der allgemeinen Anordnung und Fertigung durch 2D-Trennverfahren können viele Sonderfälle auftreten, welche im Folgenden kurz erläutert werden. Diese Fälle werden in dieser Arbeit durch die Validierung der implementierten Lösung nicht überprüft. Sie treten selten auf, sind theoretische Überlegungen oder durch eine manuelle Vorverarbeitung im Regelfall entschärft.

Holenesting: Die Eingabemenge kann auch Teile enthalten, die aus einem „Ring“ bestehen, beziehungsweise mit sehr großen Ausschnitten im Inneren versehen sind. Dies führt dazu, dass große Flächen nach dem Schnitt als Restmaterial entsorgt werden müssen, obwohl die Fläche groß genug ist, um andere Teile der Eingabemenge darin zu platzieren und damit den Gesamtverschnitt zu verringern. Das Platzieren von Geometrien innerhalb anderer Geometrien kann auch vor dem eigentlichen Platzierungsvorgang durchgeführt werden. Damit reduziert sich zusätzlich die Menge der zu platzierenden Geometrien, da nur die Außenkonturen für eine Platzierung relevant sind.

Perfect Fit: Im Anwendungsfall eher selten anzutreffen ist die Situation, in welcher ein zur Verfügung stehender Raum durch die Eingabemenge vollkommen ausgenutzt werden kann. Es handelt sich dabei, wie bei einem Puzzle, um ein eher theoretisches Problem, welches die Mächtigkeit von Lösungsalgorithmen in Bezug zu der Lösungsgenauigkeit und Variantenvielfalt aufzeigt. Im Anwendungsfall ist der Sonderfall am ehesten Anzutreffen, wenn aus einer definierten Fläche die maximale Anzahl der gleichen Geometrie geschnitten werden soll.

Formencharaktaristika: In vielen Anwendungsfällen häufen sich konstruktionsbedingt gleiche oder sehr ähnliche Außenkonturen von zu fertigenden Teilen. Eine solche Häufung ist vor allem bei komplexeren Konturen schwer automatisch zu erfassen und sinnvoll weiterzuverwenden. Bei einfacheren Konturen wie Ähnlichkeit mit Kreisen, Rechtecken oder anderen primitiven Geometrien können über Heuristiken Platzierungsoperationen optimiert werden. Schöning [STTM+01] beispielsweise verwendet direkt spezielle Algorithmen für den Fall von (kreisrunden) Garnspulen.

Ausreizung des Werkbereichs: Im Bereich des Prototypenbaus ist es möglich, die Konstruktion an die bestehende Fertigungsmaschine anzupassen. Es handelt sich bei den Sonderfällen um Großteile, welche die Maße des Rohmaterials in der Längendimension ausnutzen und sich nur in sehr wenigen oder genau einer Position anordnen lassen. Dies führt meist zu komplexen und schlecht nutzbaren Reststücken. Bei einer automatischen Platzierung erzeugt das eine sehr hohen Fehlerquote und sollte nicht direkt als Simulationseingabe genutzt werden.

6.6 Testfälle

Die konkret durchgeführten Testfälle mit den soeben beschriebenen Eingabedaten werden im Folgenden näher erläutert. Sie sind eine Auswahl aus den vorgestellten Testdaten.

6.6.1 Benchmarks Die gewählten Benchmarks passen von Teileumfang und Komplexität zum vorliegenden Anwendungsgebiet und bieten auch hinsichtlich der Komplexität verwandte Außenkonturen wie reale Fertigungsdaten. Es ist dabei zu erwarten, dass eine

Testfall	Typ	Geometrien	Gesamtanzahl Objekte	∅ Eckpunkte	Komplexität
blaz1	Benchmark	7	28	6,29	mittel
blaz2	Benchmark	4	20	7,5	einfach
jakobs1	Benchmark	25	25	5,6	schwer
jakobs2	Benchmark	25	25	5,36	schwer
batch	modifiziert	7	7+7+7+7	6,29	einfach
lw	Anwendungsfall	3	14	36,6	mittel
kar	Anwendungsfall	7	7	94,85	schwer

gute Platzierung gefunden werden kann, die sich bezüglich des Nutzungsgrades des Rohmaterials mit bekannten Lösungen vergleichen lässt.

6.6.2 Modifizierte Benchmarks Der Benchmark blaz1 wurde so modifiziert, dass die 28 zu platzierenden Teile nicht in einer Charge platziert werden, sondern auf vier gleiche Datensätze verteilt sind. Alle anderen Vorgaben sind wie im originalen Benchmark umgesetzt. Die Stückelung der Eingabemenge hat zur Folge, dass Teile die in einer vorangegangen Charge bereits platziert wurden, nicht mehr verschoben werden können. Es ist dabei zu erwarten, dass ein etwas schlechteres Ergebnis als bei der ursprünglichen Berechnung erreicht wird. Dennoch ist es möglich, nah an die Ergebnisse anderer Berechnungen unter den Originaleinstellungen heranzukommen.

6.6.3 Anwendungsfälle Für den tatsächlichen Einsatz im Umfeld der Kleinserien- und Prototypenfertigung werden bereits existierende, platzierte und gefertigte Aufträge im Rahmen der Arbeit nachträglich nochmals automatisiert platziert. Da es sich um händische Platzierungen handelt, ist dabei zu erwarten, dass bessere Platzierungen gefunden werden und der Nutzungsgrad des Rohmaterials erhöht werden kann.

6.7 Durchführung

6.7.1 Algorithmenkonfiguration Es wird die in Kapitel 5.7 erläuterte Konfiguration eingesetzt. Wobei zur besseren Beobachtung des Verlaufs die maximale Anzahl der Generationen auf 350 erhöht wurde.

6.7.2 Verwendete Simulationshardware Für die Durchführung der Simulation wurden zwei verschiedene Plattformen genutzt, um die Performance im Anwendungsfall zu untersuchen. Ein Notebook als mögliches System in einem Anwendungsfall als mobiles Einsatzgerät an Kleinmaschinen und ein Referenzsystem für stationäre Geräte.

Notebook
- Intel® Core™i5 CPU M520 @2,4 GHz , 3MB Cache, 2 Kerne, 4 Threads
- 6 GB Arbeitsspeicher
- 64 - Bit System / Windows 7

Desktop
- AMD Athlon™64 X2 @3 GHz , 2MB Cache, 2 Kerne, 2 Threads
- 2 GB Arbeitsspeicher
- 64 - Bit System / Windows 7

6.7.3 Kommerzielles Plugin Bei der am Laserbetrieb etablierten Prozesskette im Karlsruher Institut für Technologie wird die Vorverarbeitung in einem Grafikprogramm mit Vektorformaten durchgeführt (Corel Draw). Dazu werden Einzelteile manuell hinzugefügt und auf einer Arbeitsfläche platziert, die an die Maße des Werkzeugbetts angelehnt ist. Neben der manuellen Anordnung steht zur automatischen Anordnung das kommerzielle Plugin eCut zur Verfügung. Dieses ermöglicht es über mehrere Parameter eine Anordnung der markierten Teile innerhalb einer gewählten Kontur oder am Rand der gesamten Arbeitsfläche. Es wird zur Berechnung von Vergleichswerten verwendet. Zur Durchführung wurde im Programm CorelDraw X5 folgender Ablauf eingehalten:

1. Auruf des Plugins eCut
2. Eingabeobjekte und Rohmaterialkontur importieren
3. Eingabeobjekte und Rohmaterialkontur markieren
4. Platzierungsfunktion über den Button „Nesting“ aufrufen
5. Rohmaterialkontur auswählen und als „Container“ markieren
6. Einstellungen: Für alle Abstände 0, bzw. den verfügbaren Minimalwert wählen
7. Durchführen und übernehmen der Platzierung über die Buttons „Apply“ und „Ok“

Das Ergebnis wurde als Vektordatei exportiert und anhand der selbst implementierten Zielfunktion bewertet. Dabei gilt es zu beachten, dass die Größe, bzw. Auflösung der zu platzierenden Teile eine maßgebliche Rolle spielt, da sich in der Anwendung die Mindestabstände nicht mit dem Wert 0 belegen lassen. Dies hat zur Folge, dass der Mindestabstand immer 1 der gewählten Maßeinheit ist. Dies kann die Güte der Platzierungen von niedrig aufgelösten, fiktiven Geometrien stark verfälschen.

6.7.4 Ergebnisse Nach durchgeführter Simulation wird im Folgenden eine exemplarische Auswahl an Ergebnissen näher erläutert.

Blaz1: Im ersten berechneten Benchmark ist anhand der Verlaufkurve (siehe Abb. 25) deutlich zu erkennen, wie zu Beginn schnell eine Verbesserung erreicht werden kann, dann aber eine Stagnation auftritt. An der zweiten dargestellten Kurve kann der Einfluss der randomisierten Mutationsoperationen festgestellt werden. Nach etwa 250 Generationen ohne weitere Verbesserungen kann bei Generation 320 doch noch eine günstigere Positionierung gefunden werden. Diese ist in Abbildung 26 zu sehen. Diese Lösung ist mit einer Rohmaterialausnutzung von 75,92% aber deutlich entfernt von den besten Platzierungen aktuellerer Arbeiten. Dennoch ist sie effizienter als die damalige Lösung des Benchmarkautors Jakobs.

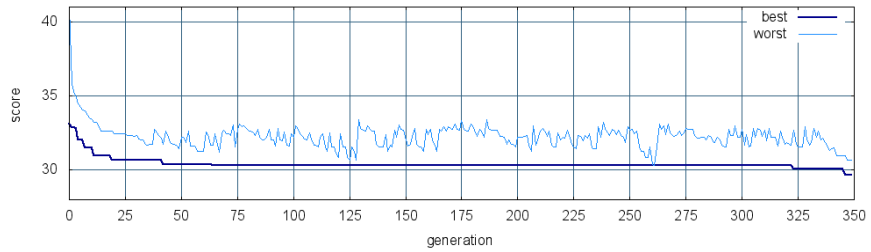

Abbildung 25. Verlauf der Bewertung von Testfall blaz1

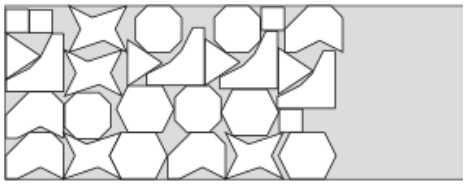

Abbildung 26. Ergebnis einer Platzierung von Testfall blaz1

Blaz2: Der zweite Benchmark ist anhand der Stückzahlen etwas überschaubarer und zeigt dies auch in seinem Bewertungsverlauf (siehe Abb. 27). Die schlussendlich verwendete Lösung wird schon kurz vor Generation 50 generiert und verbessert sich auch im weiteren Verlauf nicht mehr. Deutlich zu erkennen ist, dass durch Mutationen selbst die schlechtesten Individuen einer Population nur eine kleine Differenz zur besten Lösung besitzen. Der Ursprung dieses Effekts liegt in der kleinen Stückzahl und hohen Gleichheit der Teile untereinander, welche die maximale Anzahl an Kombinationsvarianten stark beschränkt.

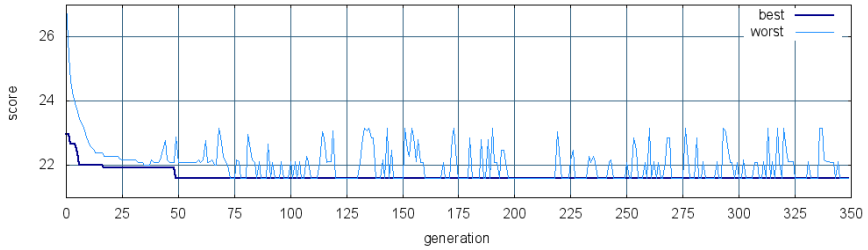

Abbildung 27. Verlauf der Bewertung von Testfall blaz2

Jakobs1: Im Verlauf (siehe Abb. 28) des Benchmarks jakobs1 ist deutlich der mehrstufige Optimierungsprozess erkennbar. Interessant ist hierbei die Betrachtung der Bewertungskurve der schlechtesten Individuen pro Population. Die deutlich sichtbare Untergrenze zeigt, dass die verwendeten Mutationen und Rekombinationen in den meisten Fällen nur Lösungen mit einer bestimmten Güte erzeugen können. Nur in sehr seltenen Zufällen erzeugen sie wirklich so gute Nachfahren, dass eine neue beste Lösung entsteht.

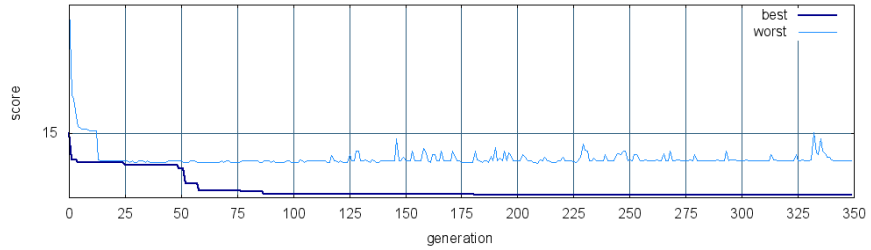

Abbildung 28. Verlauf der Bewertung von Testfall jakobs1

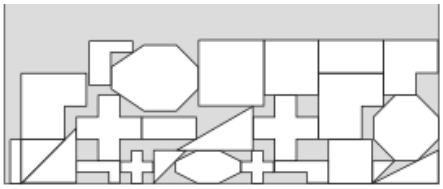

Abbildung 29. Ergebnis einer Platzierung von Testfall jakobs1

Batch: Der modifizierte Benchmark Batch (angelehnt an Blaz1) wurde in vier Durchgängen platziert, welche in Abbildung 30 gemeinsam aufgetragen sind. Es ist deutlich zu erkennen, dass bereits sehr schnell die endgültige (Teil-)Lösung gefunden wurde. Die zu permutierende Liste ist im Falle der einzelnen Platzierungen nur mit sieben Elementen besetzt, sodass hier schnell ein (lokales) Optimum gefunden wird. In der letztendlichen Platzierung (siehe Abb. 31) kann man deutlich die einzeln platzierten Chargen erkennen. Im Vergleich mit den Berechnungsergebnissen von Hopper[HoTu99], der die Platzierung in einer Charge durchgeführt hat, wird sichtbar, dass im Rahmen der Arbeit trotz der Einschränkung durch mehreren Chargen eine bessere Bewertung erreicht werden kann. Im Vergleich zu neueren Arbeiten wie der von Shalaby[ShKa13] ist jedoch ein deutliches Defizit in der Nutzungseffizienz erkennbar.

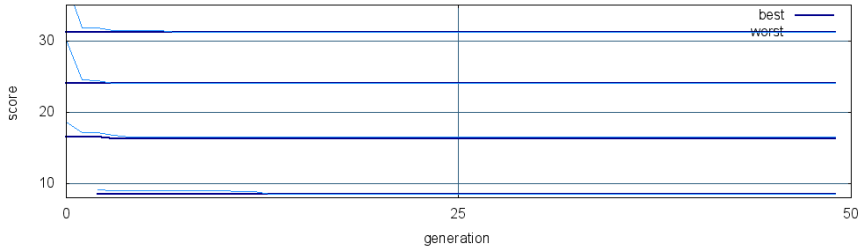

Abbildung 30. Verlauf der Bewertung von Testfall Batch

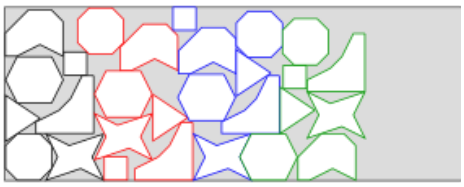

Abbildung 31. Ergebnis einer Platzierung von Testfall Batch

lw: Für die Platzierung der Konturen von Realteilen wurde auf dem Datensatz lw eine bereits bestehende händische Platzierung erneut automatisiert angeordnet. Es konnte dabei eine um 10% höhere Materialauslastung erreicht werden. In Abbildung 32 ist diese Anordnung zu sehen. Auch der Vergleich mit der zur Verfügung stehenden Anordnungssoftware zeigt, dass selbst die händische Lösung 8% platzeffizienter ist.

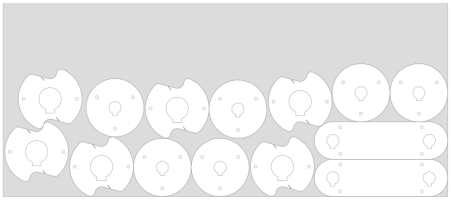

Abbildung 32. Ergebnis einer Platzierung von Testfall lw

kar: Auch beim zweiten Anwendungsfall lässt sich erkennen, dass gegenüber der händischen Lösung eine Einsparung von circa 4% erreicht werden kann. Die Breite der Teile bietet nur wenige sinnvolle Möglichkeiten der Anordnung, sodass die händische Platzierung mit etwas Überlegung gute Werte erreichen kann. Die Software eCut hingegen hat große Schwierigkeiten mit der Anordnung und erreicht nur eine Auslastung von unter 50% . In Abbildung 33 ist die beste gefundene Lösung mit einer Auslastung von 73,46% abgebildet.

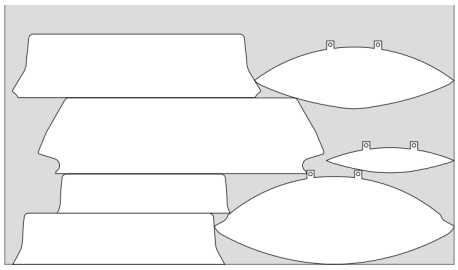

Abbildung 33. Ergebnis einer Platzierung von Testfall kar

Tabellarischer Vergleich: Für einen deutlicheren Vergleich sind die erzeugten und gesammelten Simulationsergebnisse verschiedener Arbeiten tabellarisch aufgetragen.

	2000 Hopper	2006 Burke	2009 Imamichi	2009 Umetani	2013 Shalaby	eCut [2]	händisch	2015 Fuchs	[3]
blaz1	70,42	79,41	84,25	81,68	80,9	52,07	-	75,92	90%
blaz2	63,97	74,5	-	-	74,74	52,18	-	69,98	94%
jakobs1	74,25	82,6	86,89	89,09	81,67	68,34	-	77,92	87%
jakobs2	68,4	74,8	82,51	80,84	77,2	55,61	-	74,96	91%
batch	(70,42)	(79,41)	(84,25)	(81,68)	(80,9)	-	-	71,23	85%
lw	-	-	-	-	-	57,28	65,22	76,15	100%
kar	-	-	-	-	-	46,61	69,14	73,46	100%

Lösung schlechter als beste Lösung dieser Arbeit
Beste Lösung

Abbildung 34. Auflistung der besten Werte von Vergleichsmessungen und Literaturangaben

6.8 Auswertung

Deutlich zu sehen ist im tabellarischen Vergleich, dass die hier vorliegende Arbeit in allen betrachteten Testfällen die Ergebnisse von Hopper übertreffen kann, aber zu neueren Arbeiten durchaus noch großes Verbesserungspotential besteht. Andere Ansätze und Implementierungen zeigen hier beeindruckende Ergebnisse. Leider ist es nicht möglich, diese auf neue Datensätze aus der Anwendung, wie sie zusätzlich im Rahmen der Arbeit betrachtet wurden, anzuwenden, um ihre allgemeine Tauglichkeit zu prüfen.

Die Testfälle über das im Anwendungsfall zur Verfügung stehenden Plugins eCut zeigen deutlich, wieso dieses sich im Alltagsprozess bisher nicht etabliert hat. Die generierten Lösungen basieren auf starken geometrischen Vereinfachungen. Zudem lassen sich die voreingestellten Minimalabstände nicht vollständig entfernen. Die Ergebnisse zeigen deutlich die Unterschiede zwischen theoretischen Ansätzen und den zur Verfügung stehenden generalisierten Lösungen. Da in der Praxis meist die händische Platzierung durchgeführt wird, ist der Tabelle deutlich zu entnehmen, dass hier mithilfe des entwickelten Verfahrens Einsparungen erreicht werden können.

6.8.1 Optimale Lösungen Die in der letzten Tabellenspalte angegebenen, prozentualen Anteile an der besten aufgeführten Lösung müssen dabei kritisch betrachtet werden, da der Berechnungsaufwand und der damit erreichbare Nutzungsgrad keinen linearen Verlauf darstellen.

6.8.2 Laufzeiten Nicht genau ausgewertet wurden im Rahmen der Validierung die jeweiligen Ausführungslaufzeiten der Berechnungen. Da die zur Verfügung stehenden Ergebnisse anderer wissenschaftlicher Arbeiten ohne ausreichende Informationen bzgl. Laufzeit, Simulationsplattform und Testverfahren sind, ist ein Laufzeitvergleich kaum möglich. Auch ist die zeitliche Spanne der Arbeiten sehr groß, sodass bekannte Simulationsdauern nur über Korrekturfaktoren verglichen werden können.

6.9 Produktiveinsatz

Im weiter gefassten Vergleich zu den untersuchten und bereits etablierten Lösungen bietet die aktuelle Implementierung in einigen Anwendungsfällen nur eine beschränkte Einsetzbarkeit, da die Parametrierbarkeit der Platzierungsbedingungen nicht in gleichem Maße modelliert werden kann. Für bestimmte Materialien sind beispielsweise Sicherheitsabstände zwischen platzierten Teilen notwendig, welche aktuell nicht parametriert werden können.

7 Diskussion

Im Rahmen der Validierung wurde deutlich, dass die Verwendung von klassischen Benchmarks oft keine geeignete Wahl ist, wenn es darum geht eine anwendungsnahe Lösung zu entwerfen. Die einschränkenden Randbedingungen, wie maximale Konturkomplexität und erlaubte Rotationsschritte, sind im allgemeinen Anwendungsfall nicht tragbar und darauf optimierte Lösungen aus der Forschung finden für die Anwendung weniger gute Lösungen.

Es wurde im Entwurf ein Ansatz gewählt, bei dem über eine große Breite an Lösungen über Evolution für den Anwendungsfall ausreichend gute Lösungen generiert werden. Da dies in den meisten Anwendungsfällen und Forschungsarbeiten in ähnlicher Form gelöst wird, ist dieser Ansatz in vielen Arbeiten verwendet und beschrieben. Seine Stärke liegt dabei im Regelfall bei großen, sich innerhalb unterscheidenden Eingabemengen. Da keine Vergleichswerte vorliegen, ist nach Abschluss der Arbeit die Frage offen, ob ein rechnerischer Lösungsansatz über analytische Vorgehensweisen bei sehr kleinen Eingabemengen eine bessere Performance erreichen kann. Deutlich ist aber, dass auch auf theoretischer Seite der Ansatz zielgerichteter Mutationen und Rekombinationen noch genauer untersucht werden sollte, da bisherige Arbeiten sich hier auf allgemeine Mutationen für kombinatorische Probleme stützen und damit vermutlich Potential ungenutzt bleibt.

Die Bewertung der Eignung für den Produktiveinsatz lässt sich aufgrund des Programmstands und der aktuellen Lösungsgüte nicht einfach beantworten. Dies müsste durch eine Untersuchung im Arbeitsumfeld geschehen, bei der die Berechnung in eine bestehende Prozesskette integriert ist. Da dies umfangreiche weitere Änderungen an Prozesskomponenten fordern würde, ist dies aktuell nicht möglich. Durch die Konzeption des Algorithmus auf Basis der Anforderungen und Erfahrungen im Anwendungsumfeld bietet er aber langfristig Möglichkeiten, welche durch aktuelle Komponenten nicht umsetzbar sind und für die Nutzer einen hohen Mehrwert liefern würden.

8 Zusammenfassung und Ausblick

In dieser Arbeit wurde ein Algorithmus entwickelt und dieser in einer zugehörigen Implementierung mit umgebenden Framework umgesetzt. Dieser Algorithmus erlaubt es, die im Umfeld von Kleinserien- und Prototypenbau typischen Platzierungsaufgaben intelligent zu übernehmen und um neue wertvolle Funktionen, wie der mehrfachen Platzierung, zu erweitern. Die Implementierung ist dabei auf die Funktion ausgerichtet und beinhaltet in der vorliegenden Version keine Prozessintegration und Benutzeroberfläche.

Die Funktion des Algorithmus wurde mittels Benchmarks und anwendungsorientierten Testfällen überprüft und mit bekannten Verfahren aus der Forschung verglichen. Im Rahmen der Arbeit wurde die Bewertung auf die Lösungsgüte beschränkt. Aufgrund von bereits definierten Benchmarks aus der klassischen Großindustrie oder rein fiktiven Datensätzen fällt es schwer, den Algorithmus direkt einzuordnen. Für die Validierung wurden Benchmarks aus verwandten Anwendungsfällen durchgeführt. Zudem wurden von universitärer Seite Datensätze bereitgestellt, welche Lösungen manueller Platzierungen beinhalten.

Die Leistung des entwickelten und umgesetzten Algorithmus ordnet sich daher von der Lösungsgüte im Mittelfeld der untersuchten Ansätze ein, es konnten wissenschaftliche Arbeiten übertroffen werden, wobei zu den Spitzenreitern in aktuellen Veröffentlichungen teilweise noch deutliche Defizite auftreten. Im Vergleich mit den aktuellen Lösungen im Anwendungsfall, den händischen und Plugin-Lösungen, konnte aber gezeigt werden, dass hier hohes Einsparpotential genutzt werden kann.

Die hier bearbeitete Aufgabe entspringt aus einem sehr großen Feld an Prozessen, Detailaufgaben und Teilautomatisierungen, welche im Rahmen dieser Arbeit nicht annähernd ausreichend verknüpft werden konnten, um eine direkte Verwendung im Anwendungsfall zu finden. Da der Bereich der Kleinserienfertigung, Individualfertigung und Prototypenbau gerade stark im Trend liegt, kann aufbauend auf der Arbeit die Integration in den Fertigungsablauf untersucht und durchgeführt werden. Aber auch rechnerisch sollte die Verwendung von analytischen Ansätzen genauer untersucht werden, um z.B.

im Zusammenlegen von Schnittkanten, auch über die theoretische Aufgabenstellung hinaus, eine praxistaugliche Lösung zu erstellen.

Literatur

Art 66. Richard Carl Art Jr. *An approach to the two dimensional irregular cutting stock problem.* Dissertation, Massachusetts Institute of Technology, 1966.

AVPT13. Ramón Alvarez-Valdés, Francisco Parreño und José Manuel Tamarit. A grasp/path relinking algorithm for two-and three-dimensional multiple bin-size bin packing problems. *Computers & Operations Research*, 40(12), 2013, S. 3081–3090.

BaKh14. Nikhil Bansal und Arindam Khan. Improved approximation algorithm for two-dimensional bin packing. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2014, S. 13–25.

BHKW07. Edmund K Burke, Robert SR Hellier, Graham Kendall und Glenn Whitwell. Complete and robust no-fit polygon generation for the irregular stock cutting problem. *European Journal of Operational Research*, 179(1), 2007, S. 27–49.

ChHu07. Mao Chen und Wenqi Huang. A two-level search algorithm for 2D rectangular packing problem. *Computers & Industrial Engineering*, 53(1), 2007, S. 123–136.

FoPT81. Robert J Fowler, Michael S Paterson und Steven L Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information processing letters*, 12(3), 1981, S. 133–137.

GoRe13. José Fernando Gonçalves und Mauricio GC Resende. A biased random key genetic algorithm for 2D and 3D bin packing problems. *International Journal of Production Economics*, 145(2), 2013, S. 500–510.

HeLe95. Jörg Heistermann und Thomas Lengauer. The nesting problem in the leather manufacturing industry. *Annals of Operations Research*, 57(1), 1995, S. 147–173.

Hobl15. Stephen Hobley. Privater Blog über Eigenbauprojekte., September 2015.

HoTu99. E Hopper und B Turton. A genetic algorithm for a 2D industrial packing problem. *Computers & Industrial Engineering*, 37(1), 1999, S. 375–378.

HoTu01. Eva Hopper und Brian CH Turton. A review of the application of meta-heuristic algorithms to 2D strip packing problems. *Artificial Intelligence Review*, 16(4), 2001, S. 257–300.

Jako96. Stefan Jakobs. On genetic algorithms for the packing of polygons. *European journal of operational research*, 88(1), 1996, S. 165–181.

LiHe06. Hu-yao Liu und Yuan-jun He. Algorithm for 2D irregular-shaped nesting problem based on the NFP algorithm and lowest-gravity-center principle. *Journal of Zhejiang University SCIENCE A*, 7(4), 2006, S. 570–576.

oCPa15. EURO Special Interest Group on Cutting und Packaging. Research Support - 1D, 2D, 3D Datasets and Problem Generators, Oktober 2015.

OlGF00. José F Oliveira, A Miguel Gomes und J Soeiro Ferreira. TOPOS–A new constructive algorithm for nesting problems. *OR-Spektrum*, 22(2), 2000, S. 263–284.

PiAJS15. Plácido R Pinheiro, Bonfim Amaro Júnior und Rommel D Saraiva. A random-key genetic algorithm for solving the nesting problem. *International Journal of Computer Integrated Manufacturing*, (ahead-of-print), 2015, S. 1–7.

Pohl13. Hartmut Pohlheim. *Evolutionäre Algorithmen: Verfahren, Operatoren und Hinweise für die Praxis*. Springer-Verlag. 2013.

ShKa13. Mohamed A Shalaby und Mohamed Kashkoush. A Particle Swarm Optimization Algorithm for a 2-D Irregular Strip Packing Problem. *American Journal of Operations Research*, 3(2), 2013, S. 268–278.

SiVo14. Alexandros Siasos und George-Christopher Vosniakos. Optimal directional nesting of planar profiles on fabric bands for composites manufacturing. *CIRP Journal of Manufacturing Science and Technology*, 7(3), 2014, S. 283–297.

STTM+01. Uwe Schöning, Jacobo Toran, Thomas Thierauf, Jochen Messner und Uwe Bubeck. Three Algorithms for Packaging Variable Size Bobbins. 2001.

Sysw91. Gilbert Syswerda. Schedule optimization using genetic algorithms. *Handbook of genetic algorithms*, 1991.

ThCh13. Jaya Thomas und Narendra S Chaudhari. Hybrid approach for 2D strip packing problem using genetic algorithm. In *Advances in Computational Intelligence*, S. 566–574. Springer, 2013.

Trum15. Trumpf. Offizielle Website der Trumpf Werkzeugmaschinen Deutschland Vertrieb + Service GmbH + Co. KG, September 2015.

Erklärung

Hiermit versichere ich, dass ich diese Arbeit selbständig verfasst und keine anderen, als die angegebenen Quellen und Hilfsmittel benutzt, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht und die Satzung des Karlsruher Instituts für Technologie zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet habe.

Karlsruhe, den 18.12.2015

Jonas Fuchs