

Master`s Thesis

Segmentierung des Gefäßbaumes in Computertomografie- Datensätzen des menschlichen Hals- und Kopfbereichs mittels impliziter aktiver Konturen

Arne-Michael Törsel

Betreuer: Dipl.-Inf. Karsten Rink

Abgabe: 31. 08. 2005

Studiengang: Computational Visualistics

Otto-von-Guericke-Universität Magdeburg

Fakultät für Informatik

Institut für Simulation und Grafik



Danksagung

Ich bin dankbar für das hohe Maß an Unterstützung, das ich bei der Bearbeitung meiner Master`s Thesis erfahren habe. Viele Menschen standen mir mit Rat und praktischer Hilfe zur Seite. Vornehmlich gilt mein Dank meinem Betreuer Karsten Rink, der stets Zeit für eine Diskussion oder eine gute Idee hatte. Ebenso danke ich Karin Engel für wertvolle Hinweise und Gespräche. Der gesamten Arbeitsgruppe für Bildverarbeitung und Bildverstehen am Institut für Simulation und Grafik danke ich für die Möglichkeit, zwei Vorträge zu meiner Arbeit zu halten und für die Anregungen, die in den folgenden Diskussionen entstanden. Für das eifrige Korrekturlesen des teilweise sehr unbekanntes Stoffes danke ich Heike Below, Franziska Krüger und Heike & Michael Piontek. Schließlich danke ich meiner Familie für die Unterstützung das gesamte Studium hindurch und für viele aufmunternde Worte.

Arne-Michael Törsel

August 2005

Kurzfassung

Die Extraktion des Gefäßbaums aus Computertomografiedaten ist eine anspruchsvolle Aufgabe. Oft sind die Aufnahmen relativ kontrastarm und durch Bildrauschen gestört. Blutgefäße verlaufen zudem oft dicht an Gewebe mit ähnlichem Grauwert in der Computertomografiedarstellung entlang. Dadurch ist es selbst für den Betrachter manchmal schwer ein Blutgefäß zu verfolgen und es von umliegendem Gewebe zu unterscheiden. Es existieren vielzählige Verfahren zur Extraktion des Gefäßbaums, die unterschiedliche Ansätze benutzen. Diese Arbeit untersucht die Eignung des Levelsetverfahrens zur Segmentierung des Gefäßbaums in Computertomografie-datensätzen des menschlichen Hals- und unteren Kopfbereiches. Eine Anwendung ist die Planung für Operationen zur Entfernung bösartig vergrößerter Lymphknoten. In dieser Arbeit wird ein zweistufiger Ansatz verfolgt. Zuerst wird mittels des Fast Marching Verfahrens eine grobe Schätzung der Kontur, basierend auf durch den Nutzer gesetzten Markierungen, ermittelt. Im zweiten Schritt wird diese Grobkontur durch die Nutzung der effizienten Sparse Field Levelsetmethode verfeinert. Die Ergebnisse werden mit Referenzdaten verglichen, die durch eine handkorrigierte Segmentierung ermittelt wurden.

Abstract

Extraction of the vascular tree from computer tomography data is a challenging task. Often the datasets have a low contrast and considerable image noise. Also vessels often run close to tissue with similar grey value representation. This makes it even for the human eye hard to follow the vessel and distinguish them from other tissue sometimes. There are numerous methods for vascular tree segmentation using different strategies. This thesis examines the suitability of the Levelset method for vascular tree segmentation in datasets of the neck and lower head area. An application is the surgical planning for the resection of malignant lymph nodes. As vessels often proceed close to the lymph nodes it is important to know their exact location to decide on the operation. In this work a two stage algorithm is used. Firstly, a fast, rough segmentation of the vascular tree is done starting from user supplied seed points using the Fast Marching method. Secondly, this approximation is refined using the efficient Sparse Field levelset method. The results are compared against human assisted segmentation reference datasets.

Inhaltsverzeichnis

1	Ziel der Arbeit.....	1
2	Grundlagen.....	3
2.1	Anatomie der Blutgefäße.....	3
2.1.1	In den Kopf führende Arterien.....	4
2.1.2	In den Kopf führende Venen.....	5
2.2	Die Computertomografie.....	6
2.2.1	Eigenschaften von CT-Aufnahmen.....	7
2.2.2	Darstellung von Blutgefäßen in CT-Aufnahmen.....	9
2.3	Segmentierung.....	11
2.3.1	Regiongrowing.....	12
2.3.2	Watershed-Transformation.....	12
2.3.3	Deformierbare Modelle.....	13
3	Die Levelsetmethode.....	14
3.1	Grundlagen.....	14
3.2	Die Levelsetgleichung.....	15
3.3	Konstruktion der Levelset-Evolutionsgleichung.....	17
3.3.1	Advektionskomponente.....	17
3.3.2	Krümmung.....	17
3.3.3	Bildbasierter Term.....	19
3.3.4	Kombination der Terme.....	19
3.3.5	Bestimmung der Zeitschrittweite.....	20
3.3.6	Algorithmus.....	22
3.4	Anwendungen der Levelsetmethode.....	23
3.5	Die Fast Marching Methode.....	24
3.5.1	Herleitung.....	24
3.5.2	Upwind Lösungsschema.....	25
3.5.3	Algorithmus.....	26
4	Verwandte Arbeiten.....	29
4.1	Blutgefäßextraktion.....	29
4.2	Levelsetverfahren in anderen medizinischen Anwendungen.....	31
5	Strategie und Modell.....	34
5.1	Strategie.....	34
5.2	Vorverarbeitung der Bilddaten.....	36
5.3	Intensitätsbasierter Term.....	38
5.4	Verbesserte Steuerung des Fast Marching Verfahrens.....	39
6	Effiziente Levelsetverfahren.....	41
6.1	Das Narrow Band Verfahren.....	43
6.2	Das Sparse Field Verfahren.....	44
6.2.1	Terminologie.....	45
6.2.2	Algorithmus.....	47
6.2.3	Datenstrukturen.....	47
6.2.4	Konstruktion der Schichten.....	48
6.2.5	Aktualisierung des active layer.....	49
6.2.6	Aktualisierung der Schichtzugehörigkeit.....	50
6.2.7	Aktualisierung der inactive layer.....	51
7	Implementation.....	53
7.1	Eingabedaten.....	53
7.2	Programmeffizienz.....	54
7.2.1	Speicherbedarf.....	54
7.2.2	Laufzeit.....	56
7.3	Nutzerschnittstelle.....	57

8	Evaluation.....	59
8.1	Vergleichsverfahren.....	60
8.2	Ergebnisse.....	61
9	Zusammenfassung und Ausblick.....	63
9.1	Erreichtes.....	63
9.2	Probleme.....	63
9.2.1	Auslaufen der Kontur.....	63
9.2.2	Feine Strukturen.....	64
9.3	Ausblick.....	64
10	Quellen.....	66
10.1	Links.....	71
11	Inhaltsverzeichnis der CD.....	72
12	Eigenständigkeitserklärung.....	73

Abbildungsverzeichnis

Abbildung 1: Schicht einer Computertomografie-Aufnahme (unterer Halsbereich)	2
Abbildung 2: Schematische Darstellung der in Hals und Kopf führenden Arterien (Frontalsicht)	4
Abbildung 3: Schematische Darstellung der in den Kopf führenden Venen (Frontalsicht)	5
Abbildung 4: Röntgenabsorptionsvermögen von Blut und Körpergewebetypen im Vergleich	8
Abbildung 5: Schnitt der Aufnahmeebene bei unterschiedlichen Winkeln der Blutgefäße zur Ebene	9
Abbildung 6: Blutgefäße in einer CT-Datensatzschicht	10
Abbildung 7: Einbettung einer kreisförmigen Kontur in einen Kegel	14
Abbildung 8: Verschiedene Krümmungen an einem Punkt abhängig von der Richtung	18
Abbildung 9: Möglicher Verlauf eines Blutgefäßes durch eine CT-Aufnahmeschicht	33
Abbildung 10: Datenfluss im Verfahren	37
Abbildung 11: Evolution eines Levelsets innerhalb des Narrow Band Bereiches	42
Abbildung 12: Wertebereiche der Schichten	45
Abbildung 13: Ablauf des Schichtwechsels	50
Abbildung 14: Kachelung eines zweidimensionalen Bereichs	54
Abbildung 15: Screenshot	57
Abbildung 16: Segmentierte Verzweigung (innere Jugularvene)	58
Abbildung 17: Segmentierungsfehler (Auslaufen der Kontur)	63

1 Ziel der Arbeit

Das Ziel der Arbeit ist die automatische Extraktion der Blutgefäße in Computertomografie-Schichtdatensätzen des Hals- und unteren Kopfbereichs. Dazu werden implizite aktive Konturen, ein flexibles Verfahren, das zur Segmentierung in der Bildverarbeitung eingesetzt werden kann, benutzt. Das Ergebnis dieser Extraktion ist eine automatisch generierte, möglichst vollständige Markierung der Blutgefäße in den Eingabedaten, aus der Position, Ausdehnung und Topologie abgeleitet werden können.

Eine Anwendung für die so erhaltenen Daten ist die Planung von Operationen zur Entfernung der Lymphknoten im Halsbereich. Diese Operation wird als neck dissection¹ oder im Deutschen als Halsdissektion bezeichnet. Ziel ist es, bösartig vergrößerte Lymphknoten (maligne Lymphome) zu entfernen. Es gibt mehrere Formen der neck dissection (radikale, funktionelle, selektive, ...), die sich darin unterscheiden, in welchem Ausmaß Lymphknotengruppen und umliegendes Gewebe wie Muskeln und Blutgefäße entfernt werden. Für diese Operationen ist es vorbereitend nötig, die Lage der Blutgefäße im Halsbereich zu ermitteln, um festzustellen, ob eine Operation überhaupt möglich ist und den Ablauf eines Eingriffs zu planen. Verletzungen während der Operation, die besonders an den Halsschlagadern lebensgefährlich sind, können so besser vermieden werden.

Diese Informationen können manuell anhand der Computertomografie-Aufnahmen von einem Experten bestimmt werden. Eine manuelle Segmentierung ist jedoch sehr aufwändig, da die zahlreichen Schichten des Datensatzes durch den Experten bearbeitet werden müssen. Ein wichtiges Kriterium für die angestrebte automatisierte Segmentierung wird deshalb ein möglichst schnelles Verfahren mit geringem Bedarf an Benutzerinteraktionen sein. Übergeordnet ist jedoch das Kriterium der Robustheit gegenüber Bildstörungen und die Genauigkeit, da das Verfahren, abgesehen von jeglichem Geschwindigkeitsvorteil, die manuelle Segmentierung nur dann ersetzen kann, wenn zumindest eine nahezu ebenbürtige Genauigkeit erreicht wird. Die Genauigkeit wird in einem Evaluationsverfahren durch den Vergleich mit bereits vorhandenen Segmentierungen ermittelt.

¹ Siehe auch: http://flexicon.doccheck.com/Neck_Dissection



Abbildung 1: Schicht einer Computertomografie-Aufnahme (unterer Halsbereich)

Die Segmentierung wird mittels der Levelsetmethode, einem Verfahren, das zur Repräsentation impliziter aktiver Konturen genutzt werden kann, durchgeführt. Die Eignung für diese Anwendung, speziell im Vergleich zu traditionellen Verfahren in der Bildverarbeitung, ist zu untersuchen. Um die Genauigkeit und Robustheit zu erhöhen, sollen neben klassischen Methoden wie der Nutzung des Bildgradienten zur Steuerung des Ausbreitungsverhaltens der aktiven Kontur auch Vorwissen über die Blutgefäße (Form, Darstellung in Computertomografie-Bildern) in das Modell einfließen.

Die Arbeit gliedert sich in acht Kapitel. Zuerst werden kurz die anatomischen und technischen Grundlagen der Arbeit dargestellt und eine kurze Übersicht über Segmentierungsverfahren in der Bildverarbeitung gegeben. Es folgt eine genaue Beschreibung der in dieser Arbeit verwendeten impliziten aktiven Konturen in Kapitel drei. Im vierten Kapitel werden verwandte Arbeiten vorgestellt. Die in dieser Arbeit verwendete Strategie und das Modell behandelt Kapitel fünf. Das sechste Kapitel stellt Methoden und Algorithmen zur effizienteren Umsetzung vor. Dann wird die Implementation der erarbeiteten Strategie unter Berücksichtigung effizienter Algorithmen beschrieben. Im achten Kapitel wird der Evaluationsprozess zur Bestimmung der Qualität des Verfahrens dargestellt. Die Arbeit schließt mit einer Zusammenfassung, die den erreichten Stand sowie einen Ausblick auf noch ungelöste Probleme zeigt. Hier wird auch die generelle Eignung impliziter aktiver Konturen für die Aufgabenstellung der Arbeit diskutiert.

2 Grundlagen

2.1 Anatomie der Blutgefäße

Die Blutgefäße bilden im Körper zwei Kreisläufe, in deren Zentrum das Herz steht. Der kleine Kreislauf verläuft zwischen Lunge und Herz, der große Kreislauf versorgt den restlichen Körper mit Blut. Die Kreisläufe werden gebildet von Arterien/Arteriolen – sie transportieren Blut vom Herzen in den Körper – und Venen/Venolen, die Blut zum Herzen transportieren. Dazwischen versorgen kleine Kapillaren einzelne Zellen mit Blut. Generell verzweigen sich die Arterien im Körper in einer baumartigen Struktur in feinere Arteriolen, die sich ihrerseits in Kapillaren verzweigen. Die Kapillaren vereinigen sich zu Venolen, die sich wiederum zu Venen ebenfalls baumartig vereinigen. Zusätzlich gibt es Querverbindungen zwischen einzelnen Arterien und Venen bzw. Direktverbindungen zwischen Arterien und Venen, so genannte *Anastomosen*. Arterien und Venen treten bis auf wenige Ausnahmen (z.B. die Aorta und der Aortenbogen) paarweise auf der linken und rechten Körperseite auf.

Um ein geeignetes Modell der Blutgefäße für die Arbeit zu erstellen, sind Kenntnisse über ihre Anatomie (Erscheinung, Lage, Topologie) nötig. Wie jedoch in [1] dargestellt wird, entsprechen in einigen Fällen nur 30% der Bevölkerung der Anatomie des Lehrbuchs. Es ist also mit erheblichen Abweichungen in der Lage und Topologie der Blutgefäße von den folgenden Darstellungen der „anatomischen Norm“ zu rechnen. Eine mögliche Anomalie ist zum Beispiel ein „umgekehrter“ Aortenbogen (*arcus aortae*), der nicht nach links sondern nach rechts in den Körper absteigt. Zusätzlich variiert die Position der Venen (besonders der dicht unter der Haut liegenden) generell stärker als die Position der Arterien. Ein Modell des Gefäßbaums darf also keine zu stark beschränkende Annahmen bezüglich Lage und Topologie der Blutgefäße treffen.

2.1.1 In den Kopf führende Arterien

Abbildung 2 stellt die in den Kopf führenden Arterien dar:

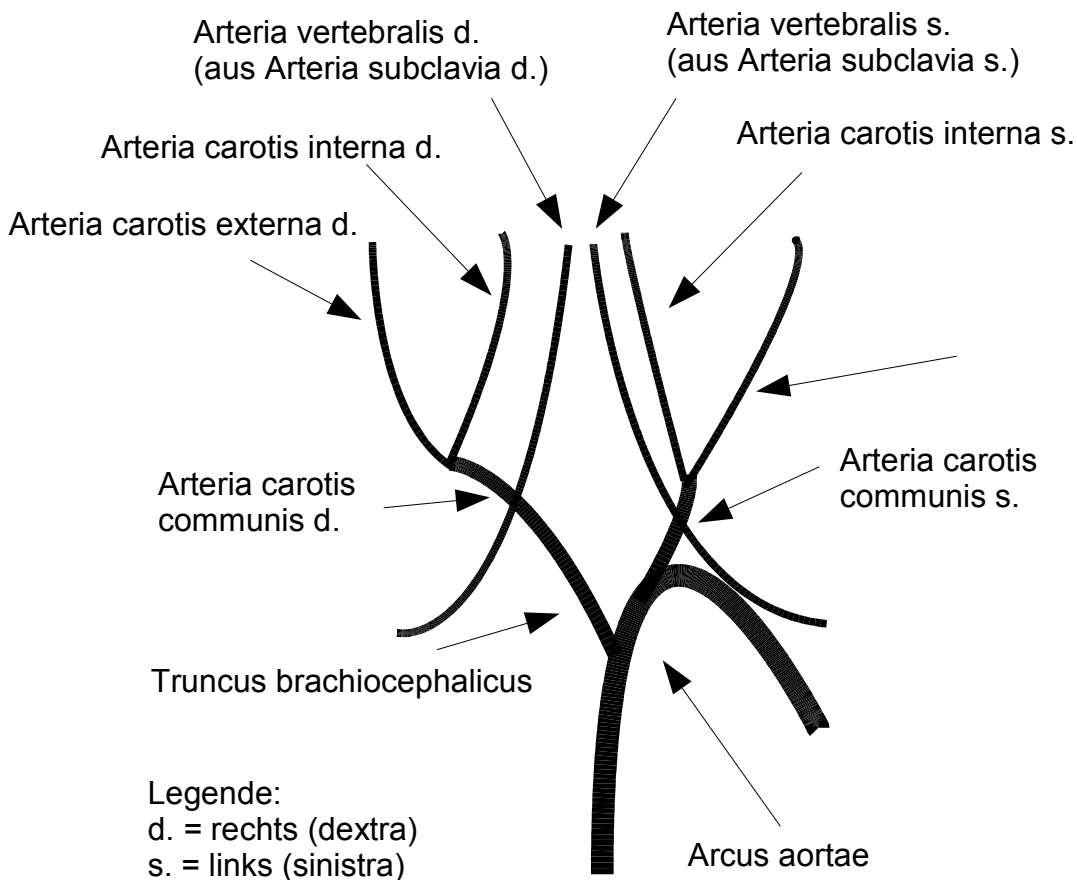


Abbildung 2: Schematische Darstellung der in Hals und Kopf führenden Arterien (Frontalsicht)

Aus dem Aortenbogen (*arcus aortae*) verzweigen zunächst die linke und rechte – über den *truncus brachiocephalicus* – Halsschlagader (*arteria carotis communis*). Die Verzweigung der Halsschlagadern ist relativ variabel (siehe auch [1] für verschiedene Verzweigungstypen und deren Häufigkeit in der Bevölkerung). Die Halsschlagadern teilen sich oberhalb des Schildknorpels² jeweils in die innere und äußere Halsschlagader (*arteria carotis interna / externa*). Zusätzlich führen die linke und rechte Wirbelsäulenschlagader (*arteria vertebralis*) aus den Schlüsselbeinschlagadern (*arteria subclavia*, nicht in der folgenden Abbildung) dicht an den Wirbelkörpern in den hinteren Schädelbereich. Dort vereinigen sie sich zur *arteria basilaris*.

Aus der äußeren Halsschlagader gehen unter anderem die

- Gesichtsschlagader (*arteria facialis*)
- Schläfenarterie (*arteria temporalis superficialis*)

² Zwei verwachsene Knorpelplatten, die den Kehlkopf bedecken

- Kieferarterie (*arteria maxillaris*)
- obere Schilddrüsenarterie (*arteria thyroidea superior*)
- Hinterhauptarterie (*arteria occipitalis*)

hervor.

Die innere Halschlagader führt in das Schädelinnere und versorgt das Gehirn und die Augen (über die Augenarterie *arteria ophthalmica*). Nach der Abgabe der Augenarterie führt die innere Halsschlagader durch die Hirnhautschichten in das Gehirn.

2.1.2 In den Kopf führende Venen

Abbildung 3 stellt die in den Kopf führenden Venen dar:

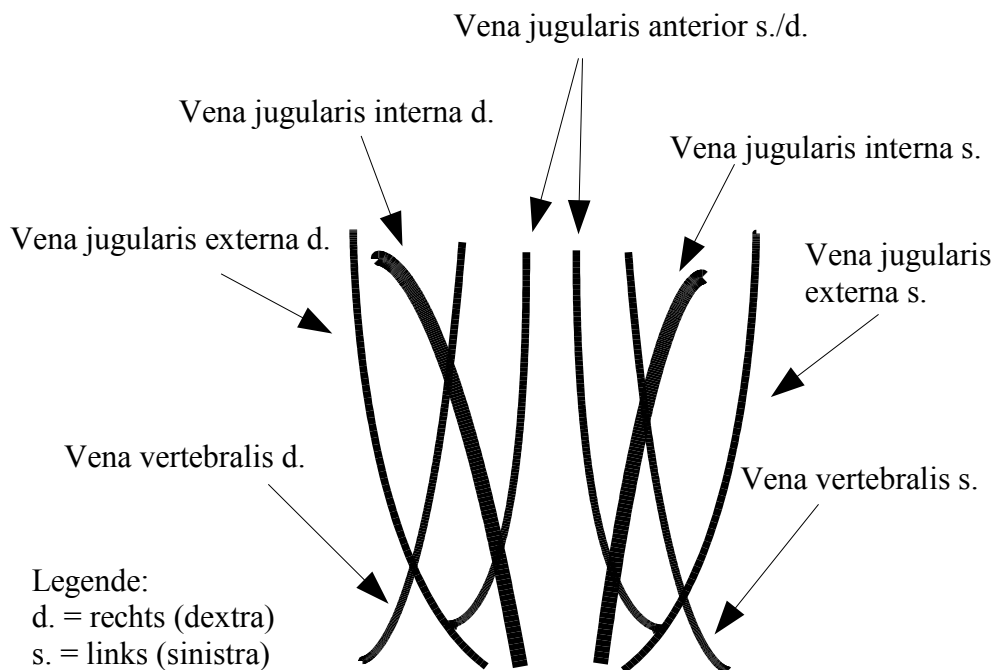


Abbildung 3: Schematische Darstellung der in den Kopf führenden Venen (Frontalsicht)

Aus dem Kopf führen die inneren, äußeren, vorderen und hinteren Jugularvenen (*vena jugularis interna, externa, anterior* und *posterior*) sowie die Wirbelsäulenvenen (*vena vertebralis*) durch den Hals zurück in Richtung Herz. Die vorderen, äußeren und inneren Jugularvenen sind dabei vielfältig durch Anastomosen untereinander verbunden.

Die innere Jugularvene verläuft zuerst dicht neben der inneren Halsschlagader (*arteria carotis interna*), dann an der Halsschlagader (*arteria carotis communis*) entlang. Sie erhält vor allem Blut aus Zuflüssen aus dem Gehirn. Die äußere Jugularvene verläuft

nahe unter der Hautoberfläche an der Seite des Halses und hat Zuflüsse aus der Gesichts- und Halsregion. Die vorderen Jugularvenen begleiten die Luftröhre und münden in den äußeren Jugularvenen. Sie transportieren ebenfalls Blut aus der Gesichts- und Halsregion. Gelegentlich tritt die vordere Jugularvene nicht als Paar, sondern einzeln als *vena mediana colli* auf. Oberhalb des Brustbeins sind die vorderen Jugularvenen durch den *arcus venosus jugularis* miteinander verbunden. Die Wirbelsäulenvenen begleiten eng den Verlauf der Wirbelsäulenschlagadern (*vena vertebralis*) an den Halswirbelkörpern, umschließen die Arterien dabei netzartig und münden in den Schlüsselbeinvenen (*vena subclavia*). Sie transportieren Blut aus Zuflüssen der Halswirbelsäulenregion.

Ein wichtiger Zufluss der inneren Jugularvene ist die Gesichtsvene (*vena facialis*), die Zuflüsse unter anderem aus der die Hinterunterkiefervene (*vena retromandibularis*) und der Schläfenvene (*vena temporalis superficialis*) erhält. Weitere Zuflüsse zur inneren Jugularvene sind die Zungenvene (*vena lingualis*) und die ein Geflecht bildenden Schlundvenen (*venae pharyngeales*).

2.2 Die Computertomografie

Die Computertomografie (kurz: „CT“) ist ein auf Röntgenstrahlen basierendes bildgebendes Verfahren, das vorwiegend in der Medizin zur Untersuchung des Körpers, aber auch in anderen Bereichen, wie zum Beispiel technischer Materialanalyse, eingesetzt wird. Die erste Aufnahme eines Menschen wurde 1971 durchgeführt. Das Verfahren ermöglicht die Analyse der inneren Beschaffenheit untersuchter Körperregionen (beziehungsweise untersuchter Objekte).

Aufgrund der unterschiedlichen Durchlässigkeit verschiedener Körpergewebe für Röntgenstrahlen lassen sich in CT-Aufnahmen Knochen sowie weiche Gewebearten voneinander unterscheiden. Zur verbesserten Abgrenzung weicher Gewebe untereinander wird gewöhnlich ein Kontrastmittel verabreicht. CT-Kontrastmittel absorbieren Röntgenstrahlen an Stellen im Körper, an denen sie sich sammeln – so werden zum Beispiel durch intravenöse Injektion eines jodhaltigen Mittels Blutgefäße und gut durchblutete Regionen besser abgegrenzt.

Eine CT-Aufnahme entsteht durch die vollständige Rotation eines Röntgenstrahlemitters (und eines Röntgenstrahldetektors auf der gegenüberliegenden Seite) um das untersuchte Objekt. Im Gegensatz zu einfachen Röntgenaufnahmen wird das Objekt also nicht nur aus einer, sondern aus allen Richtungen durchstrahlt. Ein Computer errechnet dann aus den Messungen aller Positionen ein Bild. Das gewonnene Bild ist dadurch deutlich detaillierter als eine konventionelle Röntgenaufnahme. Eine quasi dreidimensionale Darstellung der untersuchten Körperregion wird durch Stapeln vieler solcher zweidimensionaler Aufnahmen (Schichten) erreicht. Durch den Einsatz von Röntgenstrahlen besteht ebenso wie beim Röntgen ein Gesundheitsrisiko. Deshalb muss der Einsatz der Computertomografie durch die Indikation gerechtfertigt sein.

2.2.1 Eigenschaften von CT-Aufnahmen

CT-Schichtdatensätze bestehen aus einem Stapel zweidimensionaler Aufnahmen. Der Aufnahmeabstand zwischen den Schichten ist vom verwendeten Gerät beziehungsweise der verwendeten Einstellung abhängig. Generell entspricht der Bildpunktabstand zwischen zwei Schichten nicht dem Abstand innerhalb der Schicht. Der Voxel³ ist anisotrop⁴. Das Verhältnis der Voxelseitenlänge innerhalb und zwischen den Schichten unterscheidet sich dabei grob etwa in der Größenordnung einer Zehnerpotenz (also zum Beispiel 0,3 mm zu 3 mm), dadurch sind die Voxel eher streichholz- statt würfelförmig. Eine überlappende Darstellung der Schichten ist bei der Verwendung eines Spiral-Computertomografen⁵ möglich.

Für jedes Element einer Aufnahme wird der Grad der Röntgenstrahlabsorption in der *Hounsfield-Einheit* (HE) bestimmt. Stoffe mit einem höheren Absorptionsvermögen haben positive HE-Werte, die mit einem niedrigeren Absorptionsvermögen negative HE-Werte. Als Referenzwerte dienen das Absorptionsvermögen von Wasser, das mit einem HE-Wert von 0 definiert ist und Luft mit einem HE-Wert von -1000. Die Skala ist nach oben offen.

3 Kunstwort für volume picture element, bezeichnet Elemente eines dreidimensionalen Bildes

4 Die Ausdehnung ist unterschiedlich in den verschiedenen Dimensionen.

5 Die Aufnahmeeinheit rotiert dabei kontinuierlich um den sich dazu orthogonal bewegenden Patienten. Es werden spiralförmige Daten aufgenommen, aus denen durch Interpolation Schichtdatensätze gewonnen werden können.



Abbildung 4: Röntgenabsorptionsvermögen von Blut und Körpergewebetypen im Vergleich

Typischerweise sind moderne Computertomografen in der Lage, mehrere tausend Abstufungen der HE zu unterscheiden. Für eine dem Menschen verständliche Darstellung werden die HE-Werte in Klassen zusammengefasst und als Graustufenbilder dargestellt. Eine Klassifizierung ist notwendig, weil der Mensch deutlich weniger verschiedene Grauwerte als die mögliche HE-Auflösung des Computertomografen unterscheiden kann. Außerdem kann je nach Diagnoseziel die Grauwertzuordnung nichtlinear vorgenommen werden, um einen bestimmten HE-Bereich besonders detailliert darzustellen und andere Bereiche dafür auszublenden. Dieses Verfahren wird auch als *windowing* bezeichnet, weil wie durch ein Fenster nur ein Teil der Daten betrachtet wird.

Mehrere Faktoren können die Qualität einer CT-Aufnahme beeinflussen. Je nach Dosierung der Röntgenstrahlen während der Aufnahme sind die Graustufenbilder mehr oder weniger stark „verrauscht“, das heißt sie enthalten Bildstörungen. Mit glättenden Filtern (zum Beispiel einem Gauß-Filter, siehe [27]) können diese Störungen vermindert werden. Seltener treten Bewegungsartefakte auf, da eine CT-Aufnahme relativ schnell erzeugt wird. Ein weiteres Problem sind sogenannte Aufhärtungsartefakte. Sie entstehen beim Auftreffen des Röntgenstrahls auf einen Bereich hoher Röntgenabsorption, der Teile (die „weichen“ Bestandteile) des Strahlungsspektrums verringert. Dies führt zu Streifenbildung in den Aufnahmen. Ein extremer Fall sind Materialien wie Metall (zum Beispiel Zahnfüllungen) die wegen ihrer sehr hohen Röntgenabsorption zu erheblichen sternförmigen, schattenartigen Störungen in der Aufnahme führen. Moderne Computertomografen können diese Störartefakte zumindest teilweise kompensieren.

Die Abbildung der Messwerte auf Voxel stellt eine Diskretisierung der Messwerte dar. Wie bei allen Diskretisierungsverfahren kommt es deshalb auch bei der Computertomografie zu Fehlereffekten. Ein Voxel stellt stets den anteilig mittleren Absorptionswert aller im Bereich des Voxels enthaltenen Gewebetypen dar. Dieser

Effekt wird als *Partialvolumeneffekt* bezeichnet und behindert vor allem die exakte Darstellung in Relation zur Voxelgröße kleiner Strukturen. Die Auswirkungen des Partialvolumeneffekts werden deshalb mit Zunahme der Voxelgröße stärker.

2.2.2 Darstellung von Blutgefäßen in CT-Aufnahmen

Die in dieser Arbeit bearbeiteten CT-Aufnahmeschichten stellen Axialebenen⁶ des Körpers dar. Abhängig vom Schnittwinkel der Blutgefäße mit der Ebene der Aufnahmeschichten erscheinen die Blutgefäße in der Aufnahmeebene nahezu kreisförmig (orthogonal) bis stark ellipsoid/schlauchförmig (fast parallel) gestreckt.

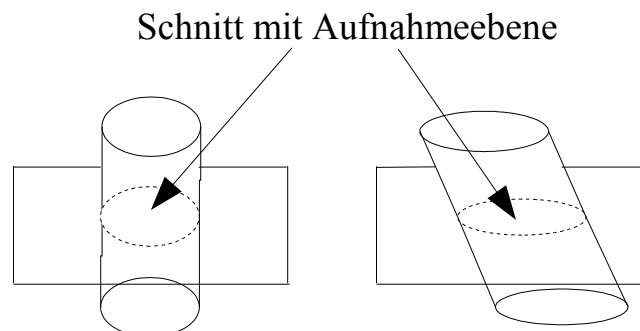


Abbildung 5: Schnitt der Aufnahmeebene bei unterschiedlichen Winkeln der Blutgefäße zur Ebene

An Gefäßverzweigungen können die auftretenden Formen stärker variieren, weil in diesem Bereich der Gefäßquerschnitt oft nicht mehr annähernd kreisförmig ist. Abhängig vom Abstand zwischen den Aufnahmen treten besonders bei einem geringen Schnittwinkel des Blutgefäßes zur Aufnahmeebene und einem großen Abstand zwischen zwei Aufnahmeschichten starke Sprünge auf, die sich in einer versetzten Darstellung äußern. Der Verlauf erscheint diskontinuierlich, so dass die Koheränz selbst durch den Menschen schwer nachvollziehbar sein kann.

Blutgefäße lagern sich oft an andere Blutgefäße oder auch Lymphknoten oder Muskeln an. Zum Beispiel verlaufen die innere Jugularvene und die Halschlagader oft direkt nebeneinander. Von einem starken Grauwertgradienten an der Rändern der Blutgefäße kann also nicht ausgegangen werden. Der Grauwertbereich der Darstellung ist abhängig vom gewählten Fensterbereich der CT-Aufnahme. Abhängig von der Menge des genutzten Kontrastmittels werden auch andere weiche Gewebe wie Lmyphknoten und Muskeln in einem ähnlichen Grauwertbereich abgebildet.

⁶ Ebenen die senkrecht zur Körperachse (Kopf-Füße) verlaufen
Siehe: <http://isgwww.cs.uni-magdeburg.de/cv/lehre/MedVisualization/glossar.html>

In der folgenden Abbildung ist eine Schicht einer CT-Aufnahme dargestellt. Die Blutgefäße sind mit weißen Ellipsen markiert. Die Abbildung zeigt auch den Schnitt durch eine Verzweigung eines Gefäßes in zwei kleinere Gefäße – dies ist zusätzlich markiert.

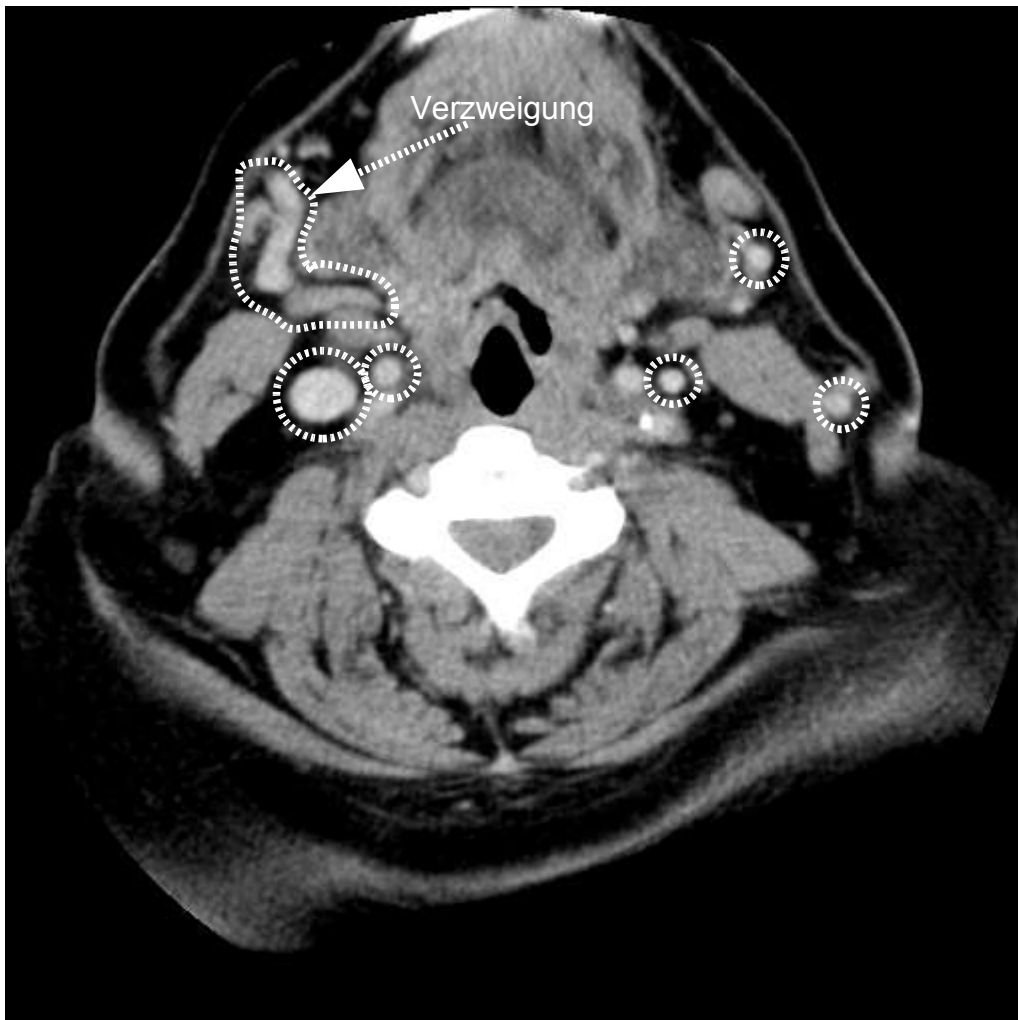


Abbildung 6: Blutgefäße in einer CT-Datensatzschicht

2.3 Segmentierung

Ziel der Segmentierung in der Bildverarbeitung ist die Extraktion von (oft semantischen) Objekten aus Bildern. Dieser für den Menschen meistens relativ einfache Prozess des „Erkennens“ von Objekten, ist für Computer oft eine anspruchsvolle Aufgabe. Der Mensch kann dabei seine Fähigkeit zur Abstraktion und das überlegene Verständnis des Kontexts nutzen. Algorithmen zur Segmentierung verwenden aus dem Bild extrahierbare Merkmale und programmiertes Vorwissen, um diese Aufgabe zu erfüllen.

Prinzipiell werden bei der Segmentierung Bildpixel disjunkten Gruppen nach definierten Kriterien zugeordnet. Es gibt viele verschiedene Verfahren, Bilder zu segmentieren, mit unterschiedlichen Stärken und Schwächen. Die Verfahren unterscheiden sich auch in der Komplexität der Implementation und Laufzeit. Die Klassifizierung der Verfahren in der Literatur ist nicht immer eindeutig. In [27] wird zum Beispiel zwischen pixelorientierten (intensitätsorientierten), regionenorientierten, kantenorientierten und modellbasierten Verfahren unterschieden.

Während die ersten drei genannten Verfahren ausschließlich lokale Bildinformationen wie Grauwerte und Gradienten verwenden, wird in modellbasierten Ansätzen versucht, ein Modell des gesuchten Objekts an einen Bildbereich anzupassen. Der modellbasierte Ansatz ist dann erfolgreicher, wenn durch Bildstörungen oder Verdeckungen in der Aufnahme das gesuchte Objekt nicht eindeutig, beziehungsweise unvollständig, dargestellt ist. Das Modell hilft hier, die Abstraktionsfähigkeit des Menschen zu imitieren, indem eine nicht hundertprozentige Anpassungsgüte des Modells akzeptiert wird. Fehlende Informationen können so ausgeglichen werden. Ein solches Modell könnte zum Beispiel eine geometrische Beschreibung des gesuchten Objekts sein. Modellbasierte Verfahren sind gewöhnlich komplexer in der Umsetzung, bieten potenziell aber robustere Ergebnisse. Ausführliche Beschreibungen verschiedener Segmentierungsverfahren finden sich in [27] und [28].

Nachfolgend werden zuerst zwei klassische Verfahren beschrieben, die für die Ziele dieser Arbeit alternativ in Frage kommen. Anschließend werden deformierbare Modelle und die in der Arbeit verwendeten impliziten aktiven Konturen erläutert.

2.3.1 Regiongrowing

Beim Regiongrowing-Verfahren wird die Segmentation ausgehend von einem Startpunkt ausgeführt. Um den Startpunkt herum wird pixelweise das Gebiet vergrößert, wenn die Nachbarnpunkte einem bestimmten Kriterium genügen. Ein einfaches Kriterium ist zum Beispiel der Grauwert eines Pixels. Da die segmentierte Fläche nur durch die Mitgliedschaft beziehungsweise den Ausschluß von Pixeln gekennzeichnet ist, ist es schwierig, geometrische Eigenschaften der Fläche und ihrer Kontur wie Krümmung und den Normalenvektor zu bestimmen. Dieser Umstand erschwert die Nutzung des Regiongrowings in Einsatzgebieten, wo solche Kriterien nötig sind, weil die Grauwert- und Gradienteninformationen nicht ausreichend für eine exakte Segmentierung sind. So ist zum Beispiel ein „Auslaufen“ der segmentierten Fläche schon dann möglich, wenn nur eine kleine „Grauwertbrücke“ zwischen zwei eigentlich disjunkten Bildbereichen besteht. Weil Blutgefäße sich oft an Gewebe mit einer ähnlichen Grauwertrepräsentation in den CT-Bildern anlagern, ist zu erwarten, dass dieser Ansatz nicht robust genug für die Ziele dieser Arbeit wäre.

2.3.2 Watershed-Transformation

Ein weiteres Verfahren zur regionbasierten Segmentierung ist die Watershed-Transformation (Wasserscheidentransformation). Die grundlegende Idee ist, die Grauwerte beziehungsweise Gradientenbeträge der Pixel als Höheninformationen, ähnlich einer topographischen Reliefkarte, zu interpretieren und einen steigenden Wasserspiegel zu simulieren. Das Wasser steigt von den lokalen Minima des Bildes in „Bassins“, ein Verschmelzen von zwei Bassins wird durch das Errichten von „Dämmen“ verhindert. Die Bassins entsprechen zum Schluss dann den segmentierten Gebieten. Deshalb eignet sich das Verfahren besonders gut, wenn die zu segmentierenden Gebiete relativ homogene Grauwerte und starke Kanten an den Grenzen besitzen. Bildstörungen führen zu einer Vielzahl kleiner Gebiete. Dieses Problem kann mit der hierarchischen beziehungsweise der markerbasierten Wasserscheidentransformation [28] behoben werden. Geometrische Eigenschaften der segmentierten Gebiete sind jedoch auch bei diesem Verfahren schwer zu berücksichtigen.

2.3.3 Deformierbare Modelle

Deformierbare Modelle sind ein in der Bildverarbeitung häufig eingesetztes Modell zur Objektsegmentierung. Im Gegensatz zu regionenbasierten Segmentierungsmethoden wird die extrahierte Region nicht nur über eine Menge markierter Pixel sondern über ein Modell beschrieben. Dies kann zum Beispiel eine Kontur sein, die das zu segmentierende Gebiet umschließt. Da es eine Beschreibung der Kontur gibt, können geometrische Eigenschaften der segmentierten Region einfacher abgeleitet werden als bei Verfahren wie dem Regiongrowing. Das Modell ist verformbar, um sich während des Segmentationsprozesses dem gewünschten Objekt anzupassen. Die Verformung wird durch auf das Modell wirkende Kräfte erreicht. Diese Kräfte werden generell in interne und externe Kräfte unterteilt. Interne Kräfte ergeben sich aus der Form des Modells und können zum Beispiel regularisierend oder expandierend wirken. Externe Kräfte sind Kräfte, die sich aus dem Bild ergeben – zum Beispiel die Stärke des Grauwertgradienten. Spezielles Anwendungswissen kann ebenfalls als externe Kraft modelliert werden. Die Kontur hat ihre Endposition erreicht, wenn eine Balance der wirkenden Kräfte erreicht ist.

Ein bekanntes deformierbares Modell ist das Snakes-Verfahren [29], das sehr häufig für die Segmentierung eingesetzt wird. Das Snakes-Verfahren ist ein parametrisch deformierbares Modell – die Kontur ist explizit repräsentiert. Snakes erfordern eine relativ gute Abschätzung der Startkontur. Topologieänderungen erfordern Erweiterungen des ursprünglichen Modells. Das später vorgestellte T-Snakes Modell [36] ist topologisch variabel auf Kosten einer höheren Komplexität.

Eine Alternative zur expliziten Repräsentation der Konturen sind die in dieser Arbeit verwendeten impliziten aktive Konturen. Sie lösen Probleme expliziter Verfahren, wie zum Beispiel Topologieänderungen und Anpassung auf höherdimensionale Problemräume, ohne, oder mit sehr geringem, zusätzlichen Aufwand. Das Levelset-Verfahren, das in dieser Arbeit genutzte implizite Modell, wird im folgenden Kapitel ausführlich vorgestellt.

3 Die Levelsetmethode

3.1 Grundlagen

Bei der Levelsetmethode ist die Kontur nicht explizit repräsentiert. Stattdessen wird eine n -dimensionale Kontur implizit als Einbettung in einer $(n+1)$ -dimensionalen Hyperoberfläche betrachtet. Ein *Levelset* ist die Menge aller Punkte, die den gleichen Wert der Levelsetfunktion Φ haben. Die zusätzliche Dimension wird aus der Abbildung der Distanz jedes Punktes zur n -dimensionalen Kontur konstruiert. Die Levelsetfunktion Φ liefert für jeden Punkt auf der Hyperoberfläche diesen Distanzwert. Das folgende Beispiel verdeutlicht dieses Prinzip:

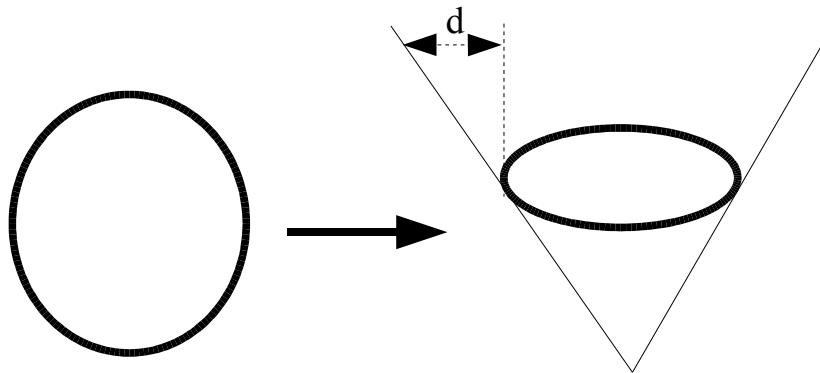


Abbildung 7: Einbettung einer kreisförmigen Kontur in einen Kegel

Das *Zero-Levelset* ist das *Levelset* für das gilt $d = 0$ und entspricht damit implizit der n -dimensionalen Kontur. Levelsetkonturen sind stets geschlossen. Das *Zero-Levelset* trennt das Innere und das Äußere der Kontur.

Mittels dieser Methode ist eine Evolution der Kontur einfach zu modellieren. Eine stark abstrahierte Möglichkeit, sich diesen Prozess vorzustellen, ist folgende: der oben erwähnte Kegel wird mit der spitzen Seite zuerst in ein Becken mit Wasser eingetaucht. An der Wasseroberfläche entsteht ein kreisförmiger „Ring“ der die Kontur darstellt. Für jeden Punkt auf der Wasseroberfläche lässt sich der Abstand d bis zur Wand des Kegels bestimmen. Genauso lässt sich für jeden Punkt innerhalb des Kegels auf dieser Ebene die Distanz bis zum Wasser bestimmen. Die Wand des Kegels trennt also ein Inneres und ein Äußeres. Nun wird der Kegel ein Stück tiefer in das Wasser getaucht. Der „Ring“, also die Kontur, ist größer geworden. Alle Distanzwerte der Punkte auf der

Wasseroberfläche und im Inneren des Kegels haben sich verändert.

Dieses Prinzip wird bei der Simulation von Konturen mit dem Levelsetverfahren benutzt: statt direkt die Bewegung der Kontur zu verfolgen, wird für alle Punkte mittels der Levelsetfunktion Φ die Veränderung ihres Abstands zum Zero-Levelset verfolgt. Praktisch wird die Modellierung einer Kontur mit dem Levelsetverfahren gegenüber expliziten Verfahren also komplexer. Im Gegenzug ergeben sich einige Vorteile. Das Verfahren behandelt problemlos Konturen, die sich während der Evolution teilen oder verschmelzen. Für das obige Beispiel könnte man sich zwei leicht überlappende Kegel vorstellen, die wiederum in das Wasserbecken getaucht werden. Anfangs sind noch zwei kreisförmige Konturen vorhanden, die bei weiterem Eintauchen aufeinander zu wachsen und schließlich zu einer Kontur verschmelzen. Dieses Verhalten ist mit expliziten Methoden nur unter zusätzlichem Aufwand zu realisieren. Ein weiterer Vorteil des Levelsetverfahrens ist die Unabhängigkeit von der Dimensionalität des Problemraumes. Das Verfahren lässt sich ohne prinzipielle Veränderungen zum Beispiel leicht von der Verfolgung von Konturen in einem zweidimensionalen Raum zur Verfolgung von dreidimensionalen Oberflächenkonturen erweitern.

Nachfolgend wird die Entwicklung einer Gleichung für die Modellierung der Evolution von Levelsets dargestellt.

3.2 Die Levelsetgleichung

Wird die initiale n -dimensionale Kontur in die $(n+1)$ -dimensionale Funktion Φ eingebettet, ergibt sich

$$\Phi(x, t=0) = d$$

wobei d der Abstand des Punktes x zur Kontur zum Zeitpunkt $t=0$ ist. Je nach Vorzeichen von d liegt der Punkt innerhalb oder außerhalb der Kontur. Die Wahl des Vorzeichens für das Innere und Äußere ist bei der Initialisierung definierbar. Für jeden Punkt, der direkt auf der Kontur s liegt ist $d = 0$.

$$\Phi(s(t), t) = 0$$

Die Punkte müssen deshalb nicht auf einer einzelnen Kontur liegen. Dies ermöglicht die Aufspaltung beziehungsweise Verschmelzung von Konturen. Das ist ein großer Vorteil der Levelsetmethode. Prinzipiell ist es möglich, jedes Levelset zu verfolgen. Meistens wird jedoch das Zero-Levelset gewählt, weil es durch den Nulldurchgang der Levelsetfunktion und der dadurch verschiedenen Vorzeichen für das Innere und Äußere der Kontur am anschaulichsten definiert ist.

Wird Φ nach t abgeleitet ergibt sich nach der Kettenregel für die Punkte auf der Zero-Levelset Kontur:

$$\Phi_t + \sum_{i=1}^N \Phi_{x_i} x_{i_t} = 0$$

wobei x_i die einzelnen räumlichen Komponenten des Frontpunktes bezeichnen. Die Konturpunkte bewegen sich in der Richtung ihrer Normalen. Die Φ_x Komponenten bezeichnen den Zufluss bzw. Abfluss an diesem Punkt. Die Front bewegt sich mit der Geschwindigkeit F , es gilt die Levelset-Evolutionsgleichung:

$$\Phi_t + F \|\nabla \Phi\| = 0$$

In einem diskreten Gitter kann die Gleichung numerisch über finite Differenzen angenähert werden und bis zur Konvergenz berechnet werden. Grundsätzlich ist das Levelsetverfahren ein lokales Verfahren. Das heißt, für jeden Punkt der Kontur wird die Lösung durch seine unmittelbare Umgebung beeinflusst. Globale Kriterien sind nur über zusätzlichen Aufwand integrierbar. Daraus ergibt sich eine Anfälligkeit für lokale Störeinflüsse.

3.3 Konstruktion der Levelset-Evolutionsgleichung

Levelsets entwickeln sich unter dem Einfluß einer Evolutionsgleichung. Innerhalb dieser Gleichung können vielfältige Kriterien kombiniert werden, um eine resultierende Kraft zu errechnen, die in Richtung des Normalenvektors der Levelsetkontur wirkt. Sethian schlägt in [10] ein passendes Modell für die Segmentierung in der Bildverarbeitung vor.

Das Modell besteht aus drei Komponenten:

- Der Advektionsterm F_A – modelliert eine konstante Kraft, die zur Ausdehnung der Kontur dient.
- Der Krümmungsterm F_G – eine Kraft, die starke lokalen Krümmungen verhindert. Sie kann die Advektionskraft unterstützen oder ihr entgegen wirken und so die Kontur „regularisieren“.
- Der bildbasierte Term k_i – repräsentiert die Informationen des Eingabebildes.

3.3.1 Advektionskomponente

Die Advektionskomponente modelliert eine konstante Transportkraft, die auf der gesamten Kontur überall mit der gleichen Kraft in Richtung des jeweiligen Normalenvektors wirkt. Sie modelliert das generelle Verhalten der Kontur – Ausdehnung oder aber auch ein Zusammenziehen. Die Transportkraft der Advektionskomponente ist ein definierbarer Programmparameter. Ein numerisches Verfahren muss ein Upwindschema nutzen. Die Nutzung zentraler Differenzenschemen würde bei nicht differenzierbaren Diskontinuitäten der Front (sogenannte „shocks“), zum Oszillieren der Front führen [3].

3.3.2 Krümmung

Dieser Term wird innerhalb der diskretisierten Levelsetgleichung zur Regularisierung der Kontur benutzt, indem er entweder die Advektionskraft weiter verstärkt (konkav) oder dämpft (konvex). Damit kann ein lokal begrenztes „Auslaufen“ der Kontur, das zu einer hohen lokalen konkaven Krümmung führt, gebremst oder ganz verhindert werden.

Die Krümmung einer dreidimensionalen Oberfläche zu bestimmen, ist schwieriger als

die Bestimmung der Krümmung einer zweidimensionalen Kurve. An einem bestimmten Punkt der Oberfläche können verschiedene Krümmungen (*normal curvatures*) in den unterschiedlichen Richtungen auftreten⁷. Das Minimum und das Maximum dieser Krümmungen werden als *principal curvatures* bezeichnet.

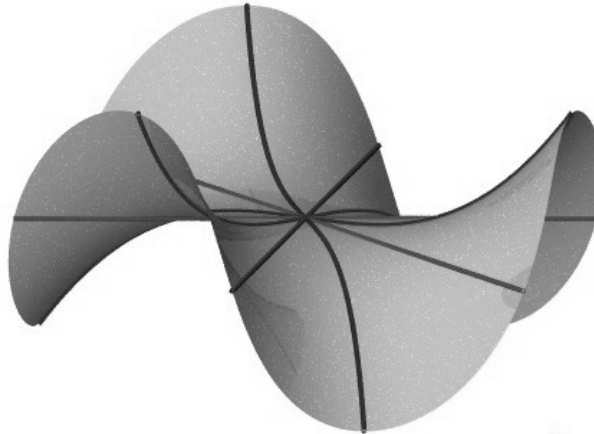


Abbildung 8: Verschiedene Krümmungen an einem Punkt abhängig von der Richtung

(Quelle:

<http://www.mathcurve.com/surfaces/selle/selle.jpg>)

Aus den *principal curvatures*, k_1 und k_2 , können nun zwei Krümmungsmetriken für den Oberflächenpunkt bestimmt werden: *mean curvature* als Mittel von k_1 und k_2 und *gaussian curvature* als Produkt von k_1 und k_2 . Die nachfolgend benutzte *mean curvature* H (mittlere Krümmung) ist also definiert als:

$$H = \frac{k_1 + k_2}{2}$$

Eine numerische Formel für die Bestimmung von H mittels zentraler Differenzen in einem diskreten dreidimensionalen Gitter ist dann:

$$H = \frac{1}{(\Phi_x^2 + \Phi_y^2 + \Phi_z^2)^{\frac{3}{2}}} \left(\Phi_{xx}(\Phi_y^2 + \Phi_z^2) + \Phi_{yy}(\Phi_x^2 + \Phi_z^2) + \Phi_{zz}(\Phi_x^2 + \Phi_y^2) - 2\Phi_x\Phi_y\Phi_{xy} - 2\Phi_x\Phi_z\Phi_{xz} - 2\Phi_y\Phi_z\Phi_{yz} \right)$$

Das Vorzeichen des Krümmungswertes ist abhängig davon, ob die Oberfläche an der entsprechenden Stelle konvex oder konkav gekrümmt ist.

⁷ Demonstrationsvideo:

<http://www.geom.uiuc.edu/zoo/diffgeom/surfspace/concepts/curvatures/prin-curv.html>

3.3.3 Bildbasierter Term

Der bildbasierte Term wirkt als Gegenkraft zur Advektionskomponente, um über das Erreichen eines Kräftegleichgewichts die Evolution der Kontur lokal zu stoppen. Er besteht aus einem gradientenbasierten und einem grauwertbasierten Teil.

Sethian schlägt in [10] folgende Gleichung für einen gradientenbasierten Term vor:

$$k_i = \frac{1}{1 + |\nabla G * I|}$$

Wobei der Ausdruck $|\nabla G * I|$ die Stärke des Grauwertgradienten in einem mit Gaußfilter G (siehe [27]) geglätteten Bild I ist. Alternativ kann auch eine Gleichung benutzt werden, die mit steigendem Bildgradienten schneller gegen Null strebt, als die erste Gleichung:

$$k_i = e^{-\alpha |\nabla G * I|}$$

Der Faktor α ist eine zur Skalierung des Terms wählbare Konstante.

3.3.4 Kombination der Terme

Ziel ist es nun, eine Evolutionsgleichung zu finden, die nahe dem angestrebten Modellzustand konvergiert. Malladi kombiniert in [10] die Advektionskomponente und den Krümmungsterm additiv. Auf diese Weise vermindert der Krümmungsterm die Advektionskomponente an Stellen konvexer Krümmung und verstärkt sie an Stellen konkaver Krümmung proportional zur Krümmungsstärke.

Der bildbasierte Term wird multiplikativ hinzugefügt. Der Grund dafür ist, dass es mit der Hinzunahme des geometriebasierten Krümmungsterms nicht mehr möglich ist, einen bildbasierten Term zu konstruieren, der bei seinem Maximum die (konstante) Advektionskomponente garantiert ausgleicht, um die Konvergenz zu erreichen. Die vorgeschlagene Evolutionsgleichung lautet also:

$$\Phi_t + k_i * (F_A + F_G) |\nabla \Phi| = 0$$

Ziel ist eine Konvergenz in der Nähe von starken Gradienten, dem Verlassen des Grauwertfensters und einer hohen, konvexen lokalen Krümmung – hier nähert sich die

wirkende Kraft also Null.

Durch die Nutzung eines Schwellwerts kann nun nach jedem Zeitschritt das Erreichen der Konvergenz getestet werden. Sei N die Anzahl der in einem Zeitschritt aktualisierten Gitterzellen und S ein Schwellwert für die mittlere absolute Veränderung des Wertes Levelsetgleichung an einem Gitterpunkt so ist Konvergenz erreicht, wenn gilt:

$$\frac{1}{N} \sum |\Delta \Phi| < S$$

3.3.5 Bestimmung der Zeitschrittweite

Die konstruierte Levelset-Evolutionsgleichung ist eine partielle Differentialgleichung, die sowohl in einem diskreten Gitter als auch in diskreten Zeitschritten numerisch gelöst wird. Die Bestimmung der optimalen Zeitschrittweite für die iterative Berechnung ist kritisch. Ist der gewählte Zeitschritt zu groß, wird das Verfahren instabil – die Front wird inkoheränt. Andererseits bedeutet ein Zeitschritt, der deutlich kleiner als nötig ist, zusätzlichen Rechenaufwand, der aufgrund des an sich aufwändigen Verfahrens ebenfalls vermieden werden muss.

Die maximal mögliche Zeitschrittgröße kann über die Courant-Friedrichs-Lewy Zahl (CFL-Zahl) errechnet werden [22]. Die Gleichung zur Berechnung der CFL-Zahl wurde ursprünglich für die numerische Strömungssimulation entwickelt. Die CFL-Zahl c gibt im diskretisierten Problemraum die maximale Änderung des Wertes einer Zelle bei einer Geschwindigkeit u , dem Zeitschritt Δt und dem Ortschritt Δx an:

$$c = \frac{u \Delta t}{\Delta x}$$

Ein numerisches Verfahren ist stabil⁸ für Werte $c < 1$. Das bedeutet für den Zeitschritt im Einheitsgitter:

$$\Delta t \leq \frac{1}{u}$$

Um die maximale Größe des Zeitschritts zu berechnen, muss die maximale Geschwindigkeit der Evolutionsgleichung ermittelt werden. Die Energieterme, die die

⁸ Die CFL-Bedingung ist notwendig, aber nicht hinreichend für die Stabilität des Verfahrens

Evolution bewirken, gehen dabei mit dem Reziprok ihrer Wichtungsfaktoren F_K für den Krümmungsterm und F_A für den Advektionsterm in die Gleichung ein. Das bedeutet, dass bei einem geringeren Wichtungsfaktor eines Terms die mögliche Zeitschrittweite steigt, weil dieser Term die Änderung des Werts der Levelsetgleichung schwächer beeinflusst.

Für den Krümmungsterm gilt im Einheitsgitter bei einer Wichtung mit dem Faktor 1 (siehe [13]):

$$\Delta t \leq \frac{1}{6}$$

Die Stärke der Advektionskomponente entspricht ihrem Wichtungsfaktor, da sie eine konstant wirkenden Kraft an allen Orten des Levelsets modelliert. Die Integration des bildbasierten Terms ist komplizierter, da er mit den anderen Termen in der Evolutionsgleichung nicht addiert sondern multipliziert wird. Sein Wert beeinflusst direkt die Wirkung der anderen Terme. Der Minimalwert des bildbasierten Terms variiert von Zeitschritt zu Zeitschritt, abhängig von den Grau- und Gradientenwerten im Bereich der Front. Der Maximalwert ist 1 – bei diesem Wert werden durch die Multiplikation die restliche Terme überhaupt nicht beeinflusst. Theoretisch ist in bestimmten Phasen der Evolution denkbar, dass dieser Maximalwert an keiner Stelle der Kontur erreicht wird. Das würde zu einer erheblichen Verlangsamung der Konturevolution in diesem Schritt führen. Damit wäre eine Erhöhung der Zeitschrittgröße in dieser Iteration möglich. Der Zeitschritt könnte dann für jeden neuen Schritt angepasst werden, indem zuerst der höchste auftretende Wert des bildbasierten Terms in diesem Schritt bestimmt wird. Werden die errechneten Werte des bildbasierten Terms gespeichert, können sie in der dann tatsächlich folgenden Auswertung der gesamten Evolutionsgleichung wiederwendet werden, um eine doppelte Berechnung zu vermeiden. In dieser Arbeit wird jedoch vom Maximalwert 1 des bildbasierten Terms für die Berechnung des Zeitschritts ausgegangen, weil eine Speicherung der Werte in einem Gitter oder einer Liste viel zusätzlichen Speicher verbrauchen würde. Der bildbasierte Term fällt also aus der Gleichung für die Berechnung der Zeitschrittgröße. Für die Größe des Zeitschritts gilt dann:

$$\Delta t \leq \frac{1}{6} \frac{1}{(F_K * F_A)}$$

Die Wahl des Zeitschritts beeinflusst die Größe des Konvergenzkriteriums k . Es gilt:

$$k = \Delta t K$$

Wobei K die für die Konvergenz angestrebte maximale durchschnittliche Zellenwertänderung pro Zeiteinheit (1) ist.

3.3.6 Algorithmus

Das Levelsetverfahren kann mit einem einfachen, aber sehr ineffizienten, Algorithmus umgesetzt werden. Dieser Ansatz wird auch als *dense field* Verfahren bezeichnet, weil alle Levelsets im Problemraum verfolgt werden.

Dense Field Levelset Algorithmus

1. Initialisiere geschlossene Startkontur, die den Problemraum eindeutig in konturinnere und -äußere Regionen teilt
2. Berechne eine vorzeichenbehaftete Distanztransformation auf dem Gitter in das Innere und das Äußere der Startkontur
3. Solange das Konvergenzkriterium nicht erreicht ist:
 4. Berechne $\Delta\Phi$, die Änderung des Wertes der Levelsetgleichung, auf allen Gitterpunkten mit einer numerischen Näherung der Evolutionsgleichung
 5. Aktualisiere das Gitter mit den neuen Werten

Schritt vier ist für die Geschwindigkeit des Algorithmus kritisch. Da die Evolutionsgleichung in jedem Schritt auf dem gesamten Problemraum ausgewertet wird, ist das Verfahren sehr aufwändig. Ein wesentlich effizienterer Algorithmus wird im Verlauf der Arbeit vorgestellt.

3.4 Anwendungen der Levelsetmethode

Die Levelsetmethode ist sehr vielseitig und wird in immer mehr unabhängigen Anwendungsgebieten eingesetzt. Ursprünglich wurde das Verfahren für die Modellierung von physikalischen Prozessen entwickelt. Speziell für die Simulation von Gas- und Flüssigkeitsströmungen ist die Levelsetmethode sehr gut geeignet. Eine der dazu wichtigen Eigenschaften von Levelsets ist die einfach mögliche Topologieänderung von Fronten. Diese Eigenschaft wird auch bei der Erstellung von Feuerausbreitungsmodellen genutzt. Eine weitere Anwendung auf diesem Gebiet ist die Simulation des Wachstums von Kristallen [32].

Ein wichtiges Einsatzgebiet ist auch die Bildverarbeitung. Das Verfahren wird, wie in dieser Arbeit, oft zur Segmentierung von Bildern verwendet – zum Beispiel in [10]. Es kann auch zur Bildfilterung genutzt werden, um Rauschen zu entfernen oder Konturen zu verstärken [11].

Whitaker beschreibt in [13] den Einsatz des Levelsetverfahrens zur Regularisierung und zum Morphing von dreidimensionalen Modellen sowie die Rekonstruktion von dreidimensionalen Modellen aus Mehrfachentfernungsmessungen. Hier wird die Möglichkeit genutzt, lokale geometrische Eigenschaften der Front zu ermitteln und abhängig davon die Evolution zu steuern.

Eine ungewöhnliche Anwendung führt Sethian in [7] an. Hier wird das Levelsetverfahren zum Berechnen von optimalen Wegen genutzt. Er benutzt das Beispiel eines Pianos, das bei einem Umzug durch eine Wohnung mit engen Korridoren und Türen manövriert werden muss. Mit einem Levelset-Frontpropagationsverfahren ist es möglich, einen Weg zum Zielpunkt und die nötigen Rotationen des Pianos zu berechnen. Weitere Anwendungen für das Finden optimaler Wege führt Deschamps in [23] an.

3.5 Die Fast Marching Methode

Das beschriebene Levelsetverfahren ist sehr aufwändig. Eine numerische Lösung muss durch Lösen der Levelsetgleichung auf dem gesamten Problemraum bis zur Konvergenz gefunden werden. Gilt jedoch, dass die Kontur sich von einer Startposition monoton entweder nur nach innen oder außen, nicht notwendig überall mit der gleichen Geschwindigkeit, ausbreitet, wird jeder Punkt des Problemraums genau zu einem Zeitpunkt (oder gar nicht) von der Front erreicht. Das Problem lässt sich dann wesentlich einfacher und schneller über die verwandte Fast Marching Methode lösen.

3.5.1 Herleitung

Bei bekannter Geschwindigkeit der Front an jedem Ort des Problemraums lässt sich der Zeitpunkt des Eintreffens der Front bestimmen. Das Levelsetverfahren ist dann als ein stationäres Problem formulierbar. Es gilt:

$$|\nabla T| F = 1$$

Der Gradient der Ankunftszeit T ist umgekehrt proportional zur Geschwindigkeit F der Front. Bewegt sich die Front langsam voran, ist der Ankunftszeitunterschied zwischen zwei nacheinander erreichten benachbarten Punkten im Raum hoch – und umgekehrt. Für den speziellen Fall, dass für alle Orte $F = 1$ gilt, entspricht die Lösung genau einer Distanztransformation von der Startposition.

Eine Lösung kann schrittweise aufgebaut werden, indem die Ausbreitung der Front über die Zeit simuliert wird. Da die Front sich immer nur auswärts (bzw. inwärts) bewegt, findet ein „Informationsfluss“ stets von bereits erreichten Punkten zu noch nicht erreichten Punkten statt. Sethian benutzt zur Erklärung das Beispiel des Gerüstbaus⁹: es wird auf dem Erdboden gestartet und jede Etage auf der vorher gebauten Etage (bereits erreichte Punkte) aufgebaut. Zur numerischen Lösung wird deshalb ein Schema benötigt, das zum Errechnen der Frontankunftszeit an einem Gitterpunkt nur „Informationen“ nutzt, die an bereits erreichten, benachbarten Punkten vorhanden sind. Ein solches Schema wird als *Upwind-Schema*¹⁰ bezeichnet. Ein adäquates first-order

9 Siehe: http://math.berkeley.edu/~sethian/Explanations/fast_marching_explain.html

10 „upwind“ ist im Englischen die dem Wind zugewandte Seite eines Schiffsegels, „downwind“ die abgewandte Seite

Upwind Schema für das Fast Marching Verfahren erzeugt für jede räumliche Richtung einen einseitigen first-order Differenzoperator, der den am frühesten erreichten der beiden Nachbarpunkte im Gitter auswählt.

3.5.2 Upwind Lösungsschema

Ein numerisches *Upwind-Schema* nach Rouy und Tourin (1992) zur Berechnung der Frontankunftszeit an einem Punkt u in einem zweidimensionalen kartesischen Gitter an den Koordinaten (i, j) ist:

$$\left[\max(D^{-x} u, -D^{+x} u, 0)^2 + \max(D^{-y} u, -D^{+y} u, 0)^2 \right]^{\frac{1}{2}} = \frac{1}{F_{i,j}}$$

wobei D^- und D^+ für räumliche Rückwärts- bzw. Vorwärtsdifferenzen und F für die Geschwindigkeit (beziehungsweise das Reziprok für die „Slowness“) am Gitterpunkt u stehen. Das Schema kann analog für einen höherdimensionalen Problemraum erweitert werden.

Zur Berechnung der Ankunftszeit $u_{(i,j)}$ kann dieses Schema in eine quadratische Gleichung überführt werden, indem die Differenzen eingesetzt werden:

$$\begin{aligned} & \max(u_{(i,j)} - u_{(i-1,j)}, u_{(i,j)} - u_{(i+1,j)}, 0)^2 + \\ & \max(u_{(i,j)} - u_{(i,j-1)}, u_{(i,j)} - u_{(i,j+1)}, 0)^2 = \frac{1}{F_{i,j}^2} \end{aligned}$$

Durch den *max*-Operator wird für jeden der Terme der Nachbarpunkt des Punktes $u_{(i,j)}$ mit dem kleineren Zeitwert ausgewählt. Ist für einen Nachbarpunkt noch kein Wert für die Ankunft der Front bestimmt, wird dieser als unendlich groß angenommen (er ist *downwind*). Sind die Ankunftszeiten beider Nachbarpunkte in einer Achsenrichtung gleich ∞ , wird dieser Term zu 0 und entfällt. Sei

$$\begin{aligned} x &= \min(u_{(i-1,j)}, u_{(i+1,j)}) \\ y &= \min(u_{(i,j-1)}, u_{(i,j+1)}) \end{aligned}$$

ergibt sich dann die quadratische Gleichung zur Berechnung von u :

$$(u - x)^2 + (u - y)^2 = \frac{1}{F^2}$$

Für den speziellen Fall, dass nur eine räumliche Differenz in die Gleichung eingeht, weil die Terme für die anderen Koordinatenachsen wie oben erwähnt entfallen, vereinfacht sich die Gleichung zu (Beispiel für x)

$$u = x + \frac{1}{F}$$

Die Konstruktion dieses Lösungsschemas und eines second order Schemas ist in [16] detailliert beschrieben.

3.5.3 Algorithmus

Ein von Sethian in [8] vorgeschlagener Algorithmus beginnt mit der Auswahl von Startpunkten für die sich ausbreitende Front. Für diese Punkte ist die Ankunftszeit der Front $T = 0$. Die Anzahl und Lage der Startpunkte ist beliebig wählbar – so ist es auch möglich, eine Strecke im Raum oder mehrere nicht zusammenhängende Startpunkte auszuwählen. Der Algorithmus zum Berechnen der Frontankunftszeiten ist:

Fast Marching Algorithmus

1. Markiere alle Startpunkte als **Alive**, $T=0$
2. Markiere alle Nachbarpunkte (4er-Nachbarschaft im 2D-Gitter) der Startpunkte, die nicht **Alive** sind, als **Trial**
3. Berechne für diese Punkte den Zeitpunkt T , an dem sie von der Front erreicht werden, nach o.a. Schema
4. Markiere alle übrigen Punkte als **Far**, $T = \infty$
5. Solange es noch als **Trial** markierte Punkte gibt:
 6. Wähle den **Trial**-Punkt mit dem geringsten Wert T
 7. Markiere diesen Punkt als **Alive**
 8. Markiere alle Nachbarn des Punkts, die als **Far** markiert sind, nun als **Trial**
 9. Aktualisiere für alle Nachbarpunkte, die **Trial** sind, den Wert T nach o.a. Upwind Schema

Kritisch für die Geschwindigkeit des Algorithmus ist Punkt 6 in der obigen Beschreibung. Aus der Menge der Punkte, deren Status *Trial* ist, muss der Punkt mit dem niedrigsten Wert für den Zeitpunkt des Erreichens T gefunden werden. Dieser Gitterpunkt kann nicht mehr durch einen Weg über noch nicht erreichte Punkte früher erreicht werden und wird deshalb als von der Front erreicht angesehen. Sethian benutzt zur Verwaltung der Menge der Trial-Punkte eine *min-heap*¹¹ Datenstruktur, in der die Werte T als Schlüssel für die Sortierung innerhalb des heap genutzt werden [8]. Der Aufwand für das Einfügen bzw. Aktualisieren eines Wertes beträgt $O(N \log N)$ bei optimaler Sortierung und bestimmt damit den Aufwand des gesamten Algorithmus, da die anderen Operationen konstanten Aufwand haben.

In [6] wird ein Verfahren aufgeführt, das den Aufwand auf $O(N)$ senken soll – hier wird argumentiert, dass die Punkte nicht exakt sondern nur annähernd nach T geordnet verarbeitet werden müssen. Dazu wird der *min-heap* durch eine sogenannte *untidy priority queue* ersetzt, die dieses Verhalten durch ein Klassifizierungsschema ermöglicht. Der Fehler verbliebe so innerhalb der Ordnung der sowieso vorhandenen Abweichung durch die numerische Annäherung mit finiten Differenzen. Aufgrund des geringen Anteils des Fast-Marching Verfahrens am Gesamtaufwand wurde das Verfahren in dieser Arbeit jedoch nicht implementiert.

¹¹ Eine Datenstruktur, in der die Elemente aufsteigend nach einem definierten Schlüsselkriterium geordnet werden

4 Verwandte Arbeiten

4.1 Blutgefäßextraktion

Es gibt viele Verfahren zur Blutgefäßextraktion die auf verschiedenen Ansätzen wie Mustererkennung und deformierbaren Modellen beruhen. Einen ausführlichen Überblick über existierende Arbeiten und eine Klassifizierung gibt [33]. Nachfolgend werden drei Verfahren zur Blutgefäßextraktion näher beschrieben.

Ein auf der Skelettierung der Blutgefäße in CT-Aufnahmen basierender Algorithmus wird in [34] vorgestellt. Ausgehend von den Grauwerten vom Nutzer gewählter Startpunkte wird ein zulässiger Grauwertbereich für Blutgefäß-Kandidatenpixel berechnet. Pixel außerhalb dieses Bereichs werden gelöscht (auf den Wert 0 gesetzt). Dann werden zwei Kopien dieser Daten angelegt. Eine Kopie wird mit einem Medianfilter ([27]) gefiltert, um die Homogenität der Grauwerte zu verstärken. Auf der zweiten Kopie wird eine Opening-Operation ([28]) ausgeführt, um Verkalkungen an den Gefäßwänden zu entfernen. Der Algorithmus berechnet nun eine Zentralachse zwischen vom Nutzer gesetzten Start- und Endpunkten und die Ausdehnung der Blutgefäße. Dazu wird eine Kostenfunktion definiert, die als Eingabe die beiden gefilterten Datensätze benutzt. Mittels der Kostenfunktion wird ein geeigneter Pfad vom Startpunkt zu den Endpunkten gesucht. Bei der Suche nach geeigneten Pfadpunkten werden die Nachbarn eines Pixels zyklisch in den drei Dimensionen untersucht. Es ergibt sich ein „schraubenförmiger“ Pfad durch das Gefäß. Entlang des gefundenen Pfades wird orthogonal nach der Gefäßgrenze gesucht, um die Zentralachse akkurat zu platzieren und den Gefäßquerschnitt abzuschätzen. Bemerkenswert ist, dass die Tests auf zwölf sehr hochauflösten Datensätzen ausgeführt wurden. Aufgrund der Auflösung waren die Voxel nahezu isotrop (bei einer Kantenlänge von ca. 0.5mm in jeder Richtung). Die Testergebnisse werden von den Autoren als akzeptabel eingeschätzt sofern die Kohärenz der Gefäße über den gesamten Datensatz gegeben ist. Konkrete statistische Ergebnisse wurden jedoch nicht angeführt. Eine Verstärkung des Kontrasts der Blutgefäßdarstellung durch Gabe eines Kontrastmittels wird vorausgesetzt.

In [25] wird der Fast Marching Algorithmus zur Segmentierung des Gefäßbaums

benutzt. Um das Verfahren robuster gegenüber „Auslaufeffekten“ zu machen, wurde die von der Front zurückgelegte Distanz überwacht und Frontabschnitte, die sehr weit vom Kopf der Front entfernt sind, eingefroren¹². Das Verfahren wurde auf MRT-Datensätzen angewendet und arbeitet sehr schnell. Zur Genauigkeit und Anzahl der Testdaten wurden jedoch keine konkreten Angaben gemacht. In [24] wird der selbe Ansatz verwendet. Tests wurden auf kontrastverstärkten MR-Angiografie-Daten mit einem Schichtabstand von 2mm ausgeführt. Die Ergebnisse sind sehr genau, es wird aber der für den Algorithmus günstige, hohe Kontrast der Blutgefäße zur Umgebung betont.

In [35] wird ein Wellenausbreitungsverfahren beschrieben, mit dem Gefäßbäume in MR-Angiografien segmentiert wurden. Eine Kostenfunktion für die Ausbreitung einer Welle wird auf der gesamten Aufnahme definiert. Die Kostenfunktion definiert einen Brechungsindex an dieser Position – ein hoher Brechungsindex bremst die Geschwindigkeit der Wellenausbreitung. Pixel, die definitiv keine Blutgefäße darstellen, bekommen einen unendlich hohen Brechungsindex, so dass sie als Barrieren für die Wellenausbreitung dienen. Nun wird die Ausbreitung einer sägezahnförmigen Welle vom Startpunkt aus simuliert. Die Wellenphase wird über benachbarte Pixel weitergegeben. Wenn die Kostenfunktion adäquat definiert ist, breitet sich die Welle im Bereich der Blutgefäße am schnellsten zu vom Nutzer gesetzten Endpunkten aus. Der Verlauf des Gefäßes wird durch eine Rückverfolgung des von der Welle zurückgelegten Weges entlang der Normalen der Wellenfronten gefunden. Dieser Ansatz ähnelt sehr dem Fast Marching Verfahren. Die Kostenfunktion wird durch die Anwendung einer Sigmoidfunktion auf dem Eingabebild berechnet. Sie bewirkt ein nichtlineares Ausblenden von Grauwerten außerhalb eines definierten, zulässigen Bereiches. Als Vorteile des Verfahrens werden seine Robustheit gegenüber lokalen, hochfrequenten Bildstörungen und die hohe Ausführungsgeschwindigkeit angeführt. Eine robuste Segmentierung setzt jedoch eine geeignete Kostenfunktion voraus, die bei nicht idealen Datensätzen schwer zu finden ist. Die nutzergesteuerte Möglichkeit, unterschiedliche Kostenfunktionen für verschiedene Bildbereiche zu definieren, wird deshalb erwähnt. Das ist jedoch für den Nutzer eine schwierige Aufgabe, weil die Wirkungsweise des Algorithmus bekannt sein muss, um geeignete Bildbereiche auszuwählen. Trotz der angeführten Robustheit des Verfahrens gegenüber Bildstörungen wird ein hoher Kontrast benötigt, weil die Kostenfunktion ausschließlich Intensitätsinformationen

¹² Das Verfahren wird auch in dieser Arbeit genutzt und in Kapitel 5.2 näher beschrieben

nutzt. Die in den durchgeführten Tests nicht segmentierten Teile lagen stets in kontrastarmen Gebieten.

4.2 Levelsetverfahren in anderen medizinischen Anwendungen

Das Levelsetverfahren wird häufig für die Segmentierung in der medizinischen Bildverarbeitung angewandt. Ein Grund dafür ist, dass anatomische Strukturen oft eine komplizierte Topologie besitzen, die ein sehr flexibles Modell erfordern. Die Levelsetmethode ist für die Segmentierung solch komplexer, topologisch variabler Strukturen gut geeignet, da sie beliebige Formen modellieren kann.

Ein mit der Zielstellung dieser Arbeit verwandtes Segmentierungsverfahren wird in [12] beschrieben. Hier werden Arterien und Venen in Magnetresonanz-Angiografien¹³ separiert. Um ein Auslaufen der Levelsetkontur zu verhindern, werden zur Initialisierung der Kontur Zentralachsen für die Arterien und Venen erzeugt. Die Generierung der Zentralachsen ist ein automatischer, überwachter Prozess, in dem der Nutzer Kontrollpunkte vorgibt, zwischen denen mittels eines Optimierungsverfahrens ein Pfad gesucht wird. Die Kosten für die Ausbreitung des Pfades sind über das Eingabebild definiert, das mit einem Filter zur Verstärkung blutgefäßartiger Strukturen behandelt wurde. Der Abstand der Kontur zur Zentralachse wird in der dann folgenden levelsetbasierten Segmentierung als Energieterm („Vesselness“-Funktion) zusätzlich zu einem intensitätsbasiertem und einem grauwertgradientenbasiertem Term benutzt.

In [18] wird ein Homogenitätsmaß beschrieben, das an Stelle der üblichen Gradienteninformationen zur Steuerung der Levelsetkontur genutzt wird. Das Maß ist ähnlich dem SUSAN Kantendetektor [30] als die Menge ähnlicher Pixel um einen Zentralpixel definiert. Das Verfahren wurde auf Magnetresonanztomografie-Aufnahmen des Kopfes zur Segmentierung der relativ gut abgegrenzten Augäpfel angewendet. Trotz der guten Abgrenzung der Augäpfel zur Umgebung in den Eingabedaten gab es gelegentliche Probleme durch „Auslaufen“ der Kontur.

Ein Verfahren zur Segmentierung des Herzmuskels in MRT-Aufnahmen wird in [19]

¹³ Die Darstellung von Blutgefäßen mittels Röntgen, Computertomografie oder Magnetresonanztomografie meistens unter Nutzung von Kontrastverstärkenden Mitteln.

beschrieben. Es besteht aus zwei Phasen. In der ersten Phase wird der Fast Marching Algorithmus zur groben Segmentierung eingesetzt. Aufbauend darauf wird die gefundene Grobkontur durch das Levelsetverfahren verfeinert. In der zweiten Phase wird die kontinuierliche Bewegung des Herzmuskels ausgenutzt. Durch ein Phasenkontrastsequenz¹⁴ genanntes, zusätzliches MRT-Verfahren ist es möglich, diese Bewegungen in Intensitätsbilder umzusetzen, aus denen Bewegungsvektoren abgeleitet werden können. Basierend auf der Divergenz der Bewegungsvektoren wurde ein Kohärenzmaß definiert – ausgehend von der Beobachtung, dass benachbarte Bewegungsvektoren an den Gewebegrenzen des Herzmuskels eine höhere Divergenz aufweisen. Innerhalb eines schmalen Bereiches wird die durch das Levelsetverfahren gefundene Kontur durch ein Optimierungsverfahren weiter verfeinert, indem Wahrscheinlichkeiten für die Zugehörigkeit eines Pixels basierend auf der Kohärenz der Bewegungsvektoren, der Intensität der Phasenkontrastsequenz und der Grauwertgradientenstärke der MRT-Aufnahmen berechnet werden. Tests wurden auf MRT-Aufnahmen von Hundeherzen ausgeführt und zeigten eine Verbesserung der Segmentierungsergebnisse der ersten Phase, insbesondere die Vermeidung von „Auslaufeffekten“.

Um die Segmentierung robuster zu machen, wurde in [20] a-priori-Wissen¹⁵ über die zu segmentierenden Objekte in das Modell integriert. Dazu wurde die typische Intensitätsverteilung als Funktion der Distanztransformation von der Objektgrenze und ein Krümmungsprofil der Objektkontur aus Trainingsobjekten gewonnen. Während des Levelsetevolutionsprozesses werden diese zusätzlichen Parameter zur Steuerung der Entwicklung genutzt. Die Integration in die Levelsetgleichung erfolgt über Terme, die bedingte Wahrscheinlichkeiten ausgehend von den Trainingsdaten ausdrücken. Das Verfahren wurde, neben synthetischen Testdaten, auf zweidimensionale MRT-Aufnahmen des Oberschenkelknochens und des corpus callosum¹⁶ angewandt. Durch die Integration der beiden globalen Metriken konnte eine höhere Unabhängigkeit von der Platzierung der Startkontur erreicht werden. Die meisten Tests wurden mit mehreren kreisförmigen Startkonturen innerhalb und außerhalb der Zielobjekte ausgeführt, in

14 Physikalische und technische Details siehe: „Cardiovascular Flow Measurement with Phase-Contrast MR Imaging: Basic Facts and Implementation” - <http://radiographics.rsna.org/cgi/content/full/22/3/651>

15 Ursprünglich in der Philosophie geprägter Begriff: bezeichnet von vornherein vorhandenes Wissen, das nicht durch Wahrnehmungen während des beschriebenen Prozesses entsteht.

16 Strangförmige Nervenverbindung zwischen beiden Gehirnhälften bei Menschen und Säugetieren.

denen die Startkonturen relativ zuverlässig zu der finalen Kontur verschmolzen. Versuche mit dreidimensionalen Daten wurden nicht durchgeführt, wobei zu vermuten ist, dass die Varianz der Intensitätsverteilung und des Krümmungsprofils bei dreidimensionalen Objekten gleicher Klasse wegen ihrer größeren Komplexität höher ist. Damit würden diese Metriken an Genauigkeit verlieren. Zu erwähnen ist auch, dass die Trainingsdaten im Fall der Oberschenkelknochen aus mehreren Nachbarschichten der jeweiligen MRT-Datensätzen, die für die Versuchsreihe verwandt wurden, gewonnen wurden. Somit wurde eine starke Korrelation zwischen Trainingsdaten und Versuchsdaten erzwungen. Generell scheint der Einsatz für eine komplexe Struktur wie den Blutgefäßbaum im Hals/Kopfbereich nicht angebracht, weil die Erscheinung zu variant und komplex ist.

Ein hybrides Modell wird in [21] zur Segmentierung von Bauchortenaneurysmen¹⁷ in Computertomografieaufnahmen verwendet. Es kombiniert ein dreiecksnetzbasierendes deformierbares Modell mit dem Levelsetverfahren. Die Knotenpunkte des Netzes werden in Richtung des Normalenvektors über die Evolutionsgleichung bewegt. Während der Ausbreitung des Modells wird das Dreiecksnetz verfeinert, um die Genauigkeit zu gewährleisten. Ziel war es, den Berechnungsaufwand des Levelsetverfahrens zu verringern und gleichzeitig die Fähigkeit des deformierbaren Modells zu erhöhen, feine Strukturen abzubilden. Das Verfahren wurde nur auf vier CT-Datensätzen getestet, zeigte dort im Mittel aber eine relativ geringe Abweichung zur manuellen Expertensegmentierung von etwa einem Voxel.

¹⁷ Erweiterung des Querschnitts von arteriellen Blutgefäßen
Siehe auch: <http://www.angiologie-online.de/Bauchortenaneurysma.htm>

5 Strategie und Modell

5.1 Strategie

Das Segmentierungsverfahren dieser Arbeit extrahiert die Blutgefäße, indem die Ausbreitung einer impliziten aktiven Kontur im Datenraum der CT-Datensätze simuliert wird. Die Ausbreitung findet im Inneren der Blutgefäße statt und stoppt an ihren Grenzen. Ausgangspunkt dafür sind Startmarkierungen im Bereich der Blutgefäße, die durch den Nutzer in den CT-Datensätzen gesetzt werden. Verzweigungen der Gefäße sollen erkannt werden. Das heisst, die Kontur muss sich dort aufspalten und die Zweige weiterverfolgen. Die Ausbreitung der Kontur erfolgt dabei in allen drei Dimensionen des CT-Schichtdatensatzes. Nur so ist es möglich, alle denkbaren Verläufe der Blutgefäße zuverlässig zu verfolgen.

Ein alternativer, nur zweidimensionaler Ansatz birgt die Gefahr, dass in bestimmten Fällen Kohärenzinformationen verloren gehen. Dieser Fall ist in der folgenden Abbildung dargestellt:

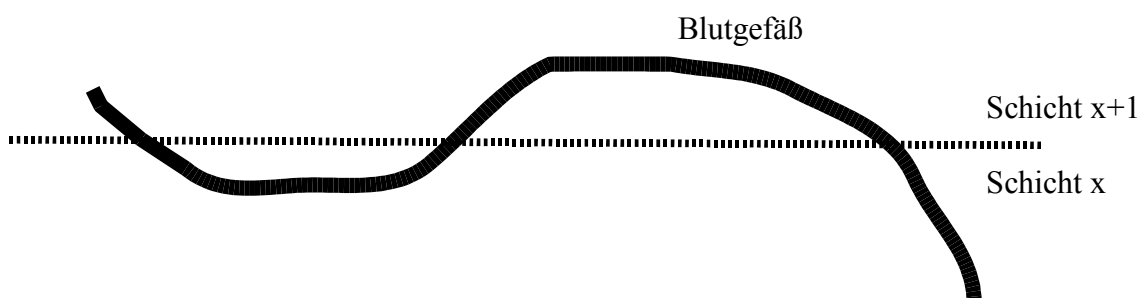


Abbildung 9: Möglicher Verlauf eines Blutgefäßes durch eine CT-Aufnahmeschicht

Ein solcher Verlauf wäre mit einem zweidimensionalen Verfahren nicht ohne Mehraufwand zu erfassen und würde eine, unter Umständen nicht eindeutig mögliche, Nachbearbeitungsphase erfordern, um die Kohärenzinformationen zu rekonstruieren.

Die für die Ausbreitung der Kontur benutzte Levelset-Evolutionsgleichung ist ein Anfangswertproblem¹⁸. Die Iterationen sind selbst mit Optimierungen wie der später beschriebenen effizienten Sparse-Field-Methode relativ aufwändig in der Berechnung.

¹⁸ Ein mathematisches Problem, das gewöhnlich aus einer Differentialgleichung besteht. Aufbauend auf dem bekannten Zustand für den Startzeitpunkt $t=0$ wird die Entwicklung des Systems berechnet. Siehe auch: <http://mathworld.wolfram.com/InitialValueProblem.html>

Eine Segmentierung der Blutgefäße unter ausschließlicher Nutzung der Levelsetmethode nur ausgehend von den Startmarkierungen ist deshalb sehr ineffizient. Es muss eine möglichst gute Schätzung für die Startkontur (Anfangswert) des Levelsetverfahrens gefunden werden, damit nur noch vergleichsweise wenige Iterationen bis zur Konvergenz benötigt werden. Dazu wird die im vorigen Kapitel beschriebene Fast Marching Methode genutzt, die eine grobe Schätzung der Endkontur liefert. Da das Levelsetverfahren die Kontur an die Grenzen der Blutgefäße expandieren soll, ist die Ausbreitungsstrategie für das Fast Marching Verfahren pessimistisch gewählt, um eine Übersegmentierung zu vermeiden.

Das Verfahren ist also wie in [4] beschrieben in zwei Phasen ausgelegt. Die erste Phase erzeugt durch eine gewollte Untersegmentierung praktisch einen „Rohbau“ der Ergebniskontur. Dieser Rohbau wird dann in der zweiten Phase durch einen Levelsetalgorithmus an die Grenzen der Blutgefäße expandiert und verfeinert. Außerdem wird die Kontur regularisiert um harte Kanten zu entfernen. Außerdem werden kleine, eingeschlossenen unsegmentierte Gebieten, die zum Beispiel durch Bildstörungen entstehen können, integriert.

Die Interaktion wird möglichst gering gehalten. Das Fast Marching Verfahren benötigt Startpunkte für die Ausbreitung. Der günstigste Fall ist, dass die Gefäße nur an ihrem Ansatz in der tiefsten Schicht des CT-Datensatzes vom Nutzer durch das Setzen von Saatpunkten innerhalb des Gefäßes markiert werden. Danach werden die markierten Gefäße automatisch unter Beachtung von möglichen Verzweigungen und Anastomosen kranial¹⁹ verfolgt. Zur Unterstützung der Segmentierung können zusätzliche Saatpunkte in höheren Schichten des Datensatzes gesetzt werden. Die sich dann von den Saatpunkten parallel ausbreitenden Konturen verschmelzen im Verlauf der Segmentierung beim Zusammentreffen zu einer Kontur.

Problematisch ist, dass aufgrund des Aufnahmeverfahrens nicht von einer klaren Abgrenzung der Darstellung der Blutgefäße von umliegenden Gewebe ausgegangen werden kann (die Magnetresonanztomografie ist für die Unterscheidung verschiedener Weichgewebetypen theoretisch besser geeignet). Solche Gewebe sind zum Beispiel die Lymphknoten oder auch Muskeln im Halsbereich wie der Kopfwendemuskel *musculus*

¹⁹ Medizinischer Fachbegriff: zum Kopf hinführend

sternocleidomastoideus. Der in beiden Phasen genutzte Grauwertgradient ist allein nicht zuverlässig genug um die Kontur zu stoppen. Deshalb werden Maßnahmen getroffen, um ein „Auslaufen“ der Kontur in der Aufnahme zu verhindern.

Um das Auslaufen der Kontur während der ersten Phase zu verhindern, wird dynamisch eine enge Grauwertfenster auf Basis der Saatpunktumgebungen erzeugt. Dies wird im folgenden Abschnitt näher erläutert. Weiterhin wird die oben beschriebene Freezing-Erweiterung für das Fast Marching Verfahren genutzt. Es unterstützt die Segmentierung langer, röhrenförmiger Objekte (wie Blutgefäße), indem Teile der Kontur bereits eingefroren werden, während der „Kopf“²⁰ der Kontur noch den Weg durch das Objekt zurücklegt. Es bestünde sonst die Gefahr, dass die Kontur nach einiger Zeit die Objektgrenzen in der Nähe des Startpunkts übertritt, bevor die Kontur das Ende des Objekts erreicht hat. Der Grund ist, dass die Grenzen als Gebiete sehr geringer Frontgeschwindigkeit definiert sind – vergeht eine zu lange Zeit, hat die Front selbst mit geringer Geschwindigkeit dieses Gebiet überwunden.

In der Levelsetphase wird ebenfalls ein Grauwertfenster genutzt. Dieses Fenster ist jedoch nicht so restriktiv wie in der Fast-Marching-Methode. Eine weitere Maßnahme ist die Beachtung der lokalen Krümmung der Kontur – eine in einem Bereich der Kontur stark konvexe Krümmung ist ein Hinweis darauf, dass die Kontur den gewünschten Bereich verlässt und wird deshalb gebremst.

Es werden nun Evolutionsgleichungen für das Fast-Marching- und das Levelsetverfahren dargestellt.

5.2 Vorverarbeitung der Bilddaten

Um das Ergebnis der Segmentierung unempfindlicher gegenüber Bildstörungen wie Rauschen zu machen, werden die Bilddaten vor dem Ablauf der Segmentierungsalgorithmen vorverarbeitet.

Die Bilddaten werden zuerst mit einem Median-Filter [27] behandelt. Der Medianfilter hat aufgrund der nötigen Sortierung der Elemente im Bereich der Maske den Nachteil,

²⁰ Teil der Front, der den längsten Weg zurückgelegt hat

deutlich aufwändiger in der Berechnung zu sein, als lineare Faltungsoperationen wie der Gaußfilter [27]. Dem gegenüber hat er den Vorteil, Kanten im Bild besser zu erhalten und nicht zu verwischen (siehe [28]). Die verwendete Maskengröße beträgt 5 x 5 Pixel. Der Medianfilter homogenisiert die Grauwerte in beieinander liegenden Bildbereichen, was eine gleichförmigere Ausbreitung der Konturen im Fast Marching und Sparse Field Algorithmus bewirkt.

Eine weitere Aufgabe dieser Vorverarbeitungsschritte ist die verstärkte Abgrenzung der zu segmentierenden Strukturen von der Umgebung, um die Ausgangsbedingungen für die Segmentierung zu verbessern. Das Ziel der Fast Marching Phase ist eine grobe, skelettartige Segmentierung, auf der die Levelsetphase aufbaut. Weil das Levelsetverfahren so konfiguriert ist, dass es die durch das Fast Marching Verfahren ermittelte Startkontur generell expandiert, ist eine Untersegmentierung erwünscht, eine Übersegmentierung muss dagegen verhindert werden.

Da im Fast Marching Algorithmus keine Forminformationen, wie die lokale Krümmung, in die Evolutionsgleichung der Kontur einfließen, muss ein Auslaufen aufgrund von „Brücken“ zu umliegenden Objekten noch besser verhindert werden als in der späteren Levelsetphase. Nach der Anwendung des Medianfilters wird deshalb auf den Eingabedaten für den Fast Marching Algorithmus zusätzlich eine Grauwerterosion durchgeführt. Dabei handelt es sich um ein Verfahren, das den Rand von hellen Bereichen (wie zum Beispiel Blutgefäßen) im Bild abträgt. Ziel der Anwendung ist es, die Abstände zwischen Strukturen mit ähnlichen Grauwerten zu vergrößern, um das Auslaufen der Kontur zu erschweren. Die zu segmentierenden Strukturen werden dabei verkleinert, es wird aber sowieso nur eine Untersegmentierung erwartet.

Weil die zu segmentierenden Blutgefäße im Kopfbereich zunehmend kleiner werden, wird die Stärke der Erosion (der Radius des Strukturelements) in den höheren, kopfnäheren Schichten des Datensatzes verringert.

Das Sparse-Field Verfahren soll die mit dem Fast Marching Algorithmus gefundene Grobkontur verfeinern. Für das Verfahren ist eine Erosion der Strukturen nicht mehr akzeptabel, weil pixelgenau segmentiert werden soll. Deshalb werden die Eingabedaten für die Levelsetphase nur mit dem Medianfilter behandelt. Der Datenfluss ist

nachfolgend grafisch dargestellt:

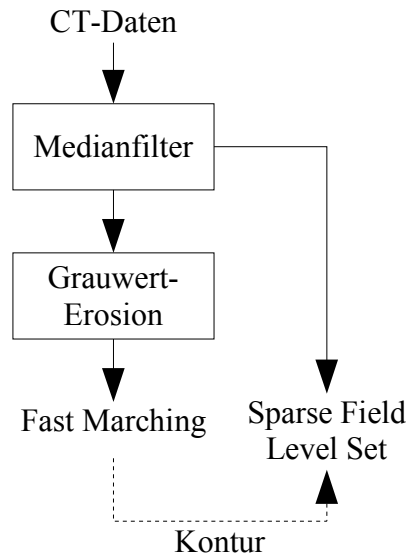


Abbildung 10: Datenfluss im Verfahren

5.3 Intensitätsbasierter Term

Für das Fast Marching und das Sparse Field Levelset Verfahren wird, wie erwähnt, zusätzlich zum Gradienten des Bildes ein zulässiger Grauwertbereich definiert. Dieser intensitätsbasierte Teil erhöht die Kraft des bildbasierten Terms, wenn der Grauwert einen bestimmten Bereich – das sogenannte Grauwertfenster – verlässt. Der Bereich wird aus einer als Programmparameter definierbaren Fensterbreite und der Inspektion der Grauwerte in der Umgebung der Saatpunkte zur Ermittlung des Bereichszentrums zu Beginn des Verfahrens erzeugt. Ein dynamischer Ansatz wurde einem fest vorgegebenen Grauwertfenster vorgezogen, weil sich die verwendeten Testdatensätze in der gewählten Abbildung von Hounsfield-Einheiten in Grauwerte (Windowing) stark unterschieden und die Datensätze unterschiedliche Kontrastmittelkonzentrationen in den Blutgefäßen aufwiesen. Ist der Grauwert außerhalb des ermittelten Fensters, wird der Betrag der Differenz zum Grauwertfenster in den bildbasierten Term integriert (bezeichnet mit P). Die Breite des Grauwertfensters ist ein experimentell ermittelter Wert. Sei C der Zentralwert dieses Grauwertfensters und t die Fensterbreite pro Richtung, dann gilt für P :

$$P = \begin{cases} 0; & \text{für } |g - C| < t \\ |g - C|; & \text{für } |g - C| \geq t \end{cases}$$

Der in der Arbeit genutzte Geschwindigkeitsterm für die Fast Marching Methode mit integrierten Grauwert- und Gradienteninformationen ist:

$$F = e^{-\alpha * (|\nabla I|^2 P^2)}$$

Die Variable I bezeichnet das mit dem Medianfilter vorverarbeitete Bild, α ist ein Skalierungsfaktor (siehe Kapitel 3.3.3). Damit die Geschwindigkeit bei steigendem Gradienten beziehungsweise Verlassen des Grauwertbereiches schneller auf 0 fällt, wurde der exponentielle Geschwindigkeitsterm basierend auf [10] gewählt. Die Kontur des Fast Marching Verfahrens soll so sensibel selbst auf geringe Gradientenbeträge und Verlassen des Grauwertbereichs reagieren.

Auch in den bildbasierten Term des Levelsetverfahrens wurde der intensitätsbasierte Term integriert. Der komplette bildbasierte Term für das Levelsetverfahren lautet dann:

$$k_i = \frac{1}{1 + |\nabla I| + P}$$

Hier wurde die reziproke Version des bildbasierten Terms aus [10] benutzt.

5.4 Verbesserte Steuerung des Fast Marching Verfahrens

Wird das Fast Marching Verfahren zur Segmentierung eingesetzt, kann das Problem auftreten, dass selbst als stark definierte Objektgrenzen, also Gebiete sehr geringer Frontgeschwindigkeit, nach einer bestimmten Zeit überwunden werden, bevor die Segmentierung komplett abgeschlossen wurde. Dieser Effekt tritt besonders dann auf, wenn die Front eine große räumliche Distanz überwinden muss. Soll zum Beispiel ein röhrenförmiges Objekt mit dem Startpunkt an einem Ende segmentiert werden, muss die Front so lange an den Grenzen um den Startpunkt herum stehen bleiben, bis das Ende der Röhre erreicht ist.

Eine von Deschamps und Cohen in [25] und [26] vorgeschlagene Erweiterung des Fast Marching Verfahrens behebt dieses Problem. Ein ähnliches Verfahren wird auch in [24] beschrieben. Es beinhaltet ein zusätzliches Einfrieren von Frontpunkten, die einen bestimmten räumlichen Abstand zum Kopf der Front überschritten haben. Dazu wird in

jedem erreichten Punkt zusätzlich der geodätische Abstand²¹ zum Startpunkt der Front gespeichert. Zu jeder Zeit ist der Maximalabstand bekannt, den die Front bereits zurückgelegt hat. Ein Element der Trial-Menge, dessen geodätischer Abstand geringer als ein definierbarer Prozentsatz der Maximalentfernung ist, wird der Alive-Menge hinzugefügt, ohne dass seine noch unerreichten Nachbarn („Far“ Elemente) in die Trial-Menge aufgenommen werden.

$$d < d_{max} T$$

Deshalb kommt es an diesen Punkt zu einem Halt der Front. Das Verfahren hat den Nachteil, dass am Ende nicht mehr gewährleistet ist, dass die Trial-Menge durch eine geschlossene Kontur von der Alive-Menge getrennt ist. Die Kontur muss deshalb nach Abschluss des Algorithmus durch ein pixelweises Contouring-Verfahren mit linearem Aufwand neu generiert werden.

²¹ Länge des kürzesten benutzbaren Weges zwischen zwei Punkten. Beim Fast Marching Verfahren bedeutet das einen Weg nur unter der Nutzung bereits erreichter Gitterpunkte.

6 Effiziente Levelsetverfahren

Selbst wenn die Initialkontur für das Levelsetverfahren bereits sehr gut durch die Fast-Marching-Methode angenähert wurde und somit nur noch relativ wenige Iterationen bis zur Konvergenz benötigt werden, ist eine Auswertung der Levelsetgleichung auf dem gesamten Problemraum sehr aufwändig. Die numerische Annäherung für die Levelsetgleichung müsste dazu in jedem Zeitschritt auf jedem diskreten Gitterpunkt ausgewertet werden. Weil das Levelsetverfahren dem eigentlichen Problem eine Dimension hinzufügt, ist der Aufwand für eine Kontur im zweidimensionalen Raum gleich $O(n^2)$ und für eine Oberfläche im dreidimensionalen gleich $O(n^3)$ – wobei n der Anzahl der Gitterpunkte pro Achse im diskretisierten Problemraum entspricht.

Ein zusätzliches Problem für die Auswertung auf dem gesamten Problemraum entsteht, wenn für die Levelsetgleichung bildbasierte Geschwindigkeitsterme genutzt werden – zum Beispiel um an starken Grauwertgradienten zu stoppen. Diese Terme sind normalerweise nur sinnvoll für den Bereich der Zero-Level-Kontur definiert, da ihre Ausbreitung mit dem Term beeinflusst werden soll. Eine Auswertung der Levelsetgleichung auf dem gesamten Problemraum erfordert jedoch eine konsistente Definition der bildbasierten Terme für alle Levelsets, die zum Beispiel verhindert, dass Levelsets untereinander interferieren.

Ein einfach zu implementierendes, aber rechenaufwändiges, Verfahren beschreiben Malladi et. al in [10]. Das Verfahren wird als *Extension* der bildbasierten Terme bezeichnet. In diesem Verfahren wird für jeden Gitterpunkt der (gemäß dem euklidischen Abstand) nächste Punkt auf dem Zero-Levelset gesucht und der Wert für den bildbasierten Term von diesem Punkt der Zero-Level-Kontur übernommen. So soll sichergestellt werden, dass die einzelnen Levelsets während der Ausbreitung nicht interferieren. Dieses Verfahren ist jedoch aus zwei Gründen aufwändig. Erstens ist es nötig, die Kontur des Zero-Levelsets in jedem Zeitschritt zu extrahieren, das heisst alle Gitterpunkte zu finden, auf denen der Nulldurchgang der Levelsetfunktion liegt. Zweitens muss für jeden Punkt ausserhalb des Zero-Levelsets der euklidische Abstand zu allen Punkten des Zero-Levelsets berechnet werden. Dieser Ansatz geht zu dem davon aus, dass zu jedem Zeitpunkt die Levelsets einen Abstand zum Zero-Levelset haben, der einer Distanztransformation entspricht. Wie [10] darstellt, ist das nicht

gegeben, wenn sich die Levelsets unter dem Einfluß von Geometrietermen wie zum Beispiel der lokalen Krümmung bewegen. In diesem Fall kann es dazu kommen, dass sich die Levelsets lokal verdichten beziehungsweise von einander entfernen.

Die Beschränkung auf die nahe Umgebung des Zero-Levelsets kann auch die Nutzung größerer Zeitschritte ermöglichen, da sich die maximale Größe des Zeitschritts nach der größten Geschwindigkeit aller Levelsets richtet. Wird nur das Zero-Levelset berechnet, muss auch nur die Geschwindigkeit seiner Evolution für die Bestimmung der Zeitschrittweite beachtet werden.

Eine Auswertung der Levelsetgleichung auf dem gesamten Problemraum ist deshalb praktisch nicht akzeptabel. Eine Möglichkeit zur Optimierung sind das nachfolgend vorgestellte *Narrow Band* Verfahren sowie das in dem Segmentierungsalgorithmus dieser Arbeit verwendete *Sparse Field* Verfahren. Die Strategie dieser beiden Methoden ist es, pro Zeitschritt nicht alle Gitterpunkte des Problemraums, sondern nur einen kleinen Teil nahe dem Zero-Levelset (oder einem anderen Levelset von Interesse) zu aktualisieren. Die Rechenkomplexität steigt daher bei den Verfahren mit der Größe der Levelsetkontur, anstatt direkt mit der Größe des Problemraums zu skalieren wie im Dense Field Ansatz.

Eine andere Möglichkeit zur Beschleunigung ist das *Adaptive Mesh Refinement* nach Milne, das in dieser Arbeit jedoch nicht weiter untersucht wird. Hier wird die Anzahl der nötigen Berechnungen ebenfalls durch eine Reduktion der betrachteten Gitterpunkte erreicht. Anstatt nur Teile eines uniform aufgelösten Gitters zu betrachten, wird ein nicht gleichmäßig aufgelöstes Gitter verwendet. Dieses Gitter hat eine hohe Auflösung an Stellen, an denen eine hohe Präzision nötig ist - zum Beispiel Konturabschnitte mit hoher lokaler Krümmung. Während der Evolution wird die Gitterauflösung dynamisch an das betrachtete Levelset angepasst.

6.1 Das Narrow Band Verfahren

Die Schlüsselidee des Narrow Band-Verfahrens ist, die Berechnungen in einer Iteration nicht auf dem gesamten Problemraum auszuführen, sondern auf einen kleinen Bereich zu beschränken, der das Zero-Levelset (oder ein anderes Levelset von Interesse) umschließt. Dieser Bereich wird als *Narrow Band* oder auch *Tube* bezeichnet. Für das Innere und Äußere des Narrow Band werden Extremwerte angenommen. Nähert sich die Kontur während der Iterationen dem Rand des Narrow Band, wird der durch das Narrow Band betrachtete Bereich neu initialisiert (die Kontur – das Zero-Levelset – innerhalb dieses Bereichs zentriert). Eine einfache Möglichkeit zur Neuinitialisierung ist die Extraktion der Kontur des Zero-Levelsets und das Berechnen der Distanztransformation für den umliegenden Problemraum. Statt der sehr aufwändigen Bestimmung der Distanzen für den gesamten Problemraum, ist es vorteilhaft, die Distanztransformation ausgehend von den Gitterzellen des Zero-Levelsets bis zur Schwelldistanz des Narrow Band zu berechnen. Das Narrow Band wird so also schichtweise um die Kontur aufgebaut. Die selbe Technik kann für die Erweiterung bildbasierter Terme angewendet werden [9]. Die Breite des Narrow Band wird heuristisch festgelegt. Sie ist ein abgeschätzter Kompromiß zwischen dem relativ hohen Aufwand der Neuinitialisierung und der Berechnung der Levelsetgleichung auf zusätzlichen Gitterpunkten im Bandbereich.

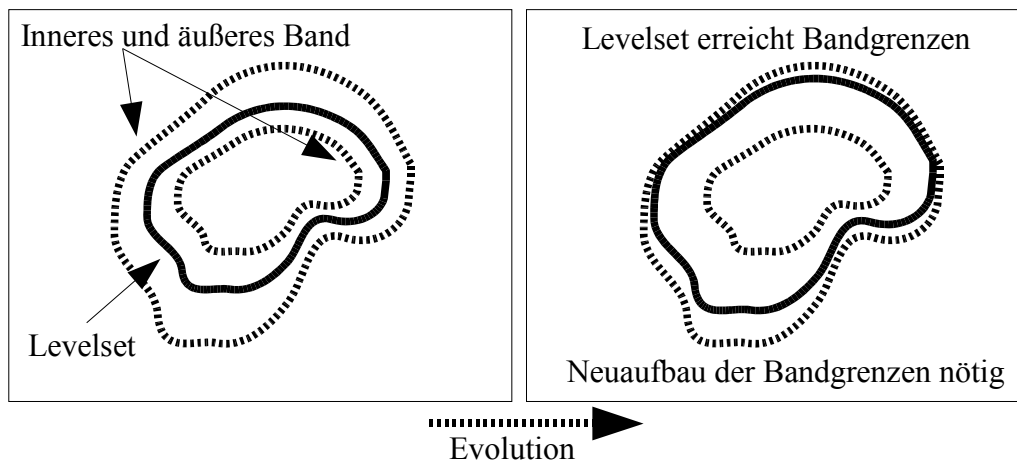


Abbildung 11: Evolution eines Levelsets innerhalb des Narrow Band Bereiches

Neben der Reduktion des Aufwands zur Berechnung hat das Narrow Band-Verfahren zwei weitere Vorteile. Erstens verringert sich der Aufwand des oben beschriebenen Verfahrens zur konsistenten Definition des bildbasierten Terms auf den Bereich innerhalb der Bandgrenzen. Zweitens ermöglicht die Beschränkung auf ein schmales

Band um die Front unter Umständen die Wahl eines größeren Zeitschrittes. Der Punkt der größten Geschwindigkeit im Problemraum erzwingt die Maximalgröße des Zeitschritts zwischen zwei Iterationen, um die Berechnung stabil zu halten (siehe 3.2.5 – Bestimmung der Zeitschrittweite). Gegebenenfalls ist innerhalb des Narrow Band eine Simulation mit größeren Zeitschritten möglich, weil die Geschwindigkeit an den Gitterpunkten kleiner ist.

6.2 Das Sparse Field Verfahren

Obwohl der Einsatz eines Narrow-Band-Verfahrens eine deutliche Effizienzsteigerung gegenüber dem Dense Field Ansatz bringt, ist es nicht optimal. In den Gitterzellen, die Teil des Narrow Band sind, werden in jedem Schritt unnötige Berechnungen außerhalb des Zero-Levelsets durchgeführt. Ein von Whitaker [14, 13] vorgeschlagener Ansatz führt in jedem Schritt nur Berechnungen für die Gitterpunkte durch, die das Zero-Levelset (den Nulldurchgang der Levelsetfunktion) enthalten. Das ist möglich, weil die Entwicklung des Zero-Levelsets nur von seiner „unmittelbaren“ Umgebung beeinflusst wird. Wird ein diskretes numerisches Verfahren zur Berechnung benutzt, ist diese Umgebung genau die Menge der Gitterpunkte, die für die Bildung stabiler räumlicher Differenzenschemen in der numerischen Evolutionsgleichung benötigt werden.

Die Strategie des Sparse Field Algorithmus ist es, in jedem Schritt die Menge der Gitterpunkte des Zero-Levelsets und der benötigten Umgebung zu verfolgen und gegebenenfalls Gitterpunkte in diese Menge aufzunehmen beziehungsweise aus ihr zu entfernen, während sich das Zero-Levelset unter der Levelsetgleichung entwickelt. Dazu trifft das Sparse Field Verfahren die vereinfachende Annahme, dass sich die Levelsets in der Umgebung in einem konstanten, gleichmäßigen räumlichen Abstand zum Zero-Levelset befinden. Das macht es möglich, die aufwändige Neuinitialisierung, die im Narrow Band Verfahren für den Aufbau des aktiven Bereichs um das Zero-Levelset benutzt wird, durch eine schnelle Näherung zu ersetzen. Weil die Umgebung des Zero-Levelsets so schnell konstruiert werden kann, entfällt der beim Narrow-Band-Verfahren nötige Kompromiß zwischen dem Aufwand zur Aktualisierung der Gitterpunkte im Narrow Band und einer Neuinitialisierung. Die Aktualisierung kann stets auf der optimalen Menge der Gitterpunkte ausgeführt werden. In jedem Zeitschritt werden nach der Aktualisierung der Gitterpunkte des Zero-Levelsets deshalb die Werte der Umgebung

mit der Näherung der Distanztransformation vom Zero-Levelset ermittelt. Experimente in [14] haben gezeigt, dass sich der dadurch entstehende Fehler innerhalb der Größenordnung des durch die finite Differenzenbildung entstehenden Fehlers bewegt und sich die absolute Genauigkeit des Verfahrens nicht verschlechtert.

Neben der größeren Effizienz besitzt das Sparse Field Verfahren die oben genannten Vorteile des Narrow-Band-Verfahrens und vergrößert sie. Eine Erweiterung bildbasierter Terme ist nicht mehr nötig, weil sie nur für Gitterzellen genutzt werden, die Teil des Zero-Levelsets sind und dort „natürlich“ definiert sind. Außerdem ist zu jedem Zeitpunkt eine diskrete Näherung der Zero-Level-Kontur vorhanden und muss nicht extrahiert werden. Das ist zum Beispiel vorteilhaft, wenn die Kontur während der Evolution grafisch dargestellt werden soll. Nachfolgend werden die Terminologie des Verfahrens und die für eine Implementation nötigen Schritte näher erläutert.

6.2.1 Terminologie

Gitterpunkte, auf denen der Nulldurchgang der Levelsetfunktion liegt, repräsentieren das Zero-Levelset. Diese Elemente werden als *active points* bezeichnet. Bei der Abbildung der Levelsetfunktion auf ein diskretes Gitter ist sichergestellt, dass die *active points* zusammen eine oder mehrere geschlossene Kontur(en) bilden - zum Beispiel in 4-er Nachbarschaft in einem zweidimensionalen Gitter. In der weiteren Beschreibung wird von einer einzelnen Kontur ausgegangen. Diese Kontur wird *active set* oder auch *active layer* genannt. Gitterpunkte in Nachbarschaft zum *active set* heißen *inactive points*. Solche Gitterpunkte lassen sich nach ihrer Entfernung nach dem Cityblock-Distanzmaß²² zum nächsten active point klassifizieren. Zusätzlich zur Distanz wird ein Vorzeichen benutzt, abhängig davon, ob der Gitterpunkt innerhalb oder außerhalb des active layer liegt. Die Menge aller inactive points mit der gleichen Distanz zur Kontur bilden wiederum eine geschlossene Kontur, die als *inactive layer* bezeichnet und durch den Distanzwert ihrer inactive points identifiziert wird. So hat der inactive layer mit dem Abstand 2 gemäß Cityblock-Distanz innerhalb der Zero-Levelset-Kontur die Nummer -2. Das Vorzeichen ist abhängig davon, ob negative Werte für das Innere oder das Äußere der Zero-Levelset-Kontur definiert sind. In dieser Arbeit werden negative Werte für das Innere und positive für das Äußere der Kontur verwendet.

Gitterzellen, die Teil des active set sind, besitzen Werte der Levelsetfunktion im Interval

²² Entspricht der Summe der absoluten Distanzen in jeder räumlichen Richtung

$[-0.5, 0.5]$. Sie repräsentieren in einem diskreten Gitter also den Ort des Nulldurchgangs der Funktion. Der Wertebereich der inactive Layer erhöht sich beziehungsweise sinkt jeweils um den Cityblock-Abstand zum active set. Das heißt, für den Layer „1“ ist der Wertebereich $]0.5, 1.5]$ und für den Layer „-1“ $[-1.5, -0.5[$. Die folgende Grafik stellt alle Wertebereiche für je zwei innere und äußere Schichten dar:

innere Zelle	Layer -2 $[-2.5, -1.5[$	Layer -1 $[-1.5, -0.5[$	Zero Layer $[-0.5, 0.5]$	Layer 1 $]0.5, 1.5]$	Layer 2 $]1.5, 2.5]$	äußere Zelle
-----------------	-------------------------------	-------------------------------	--------------------------------	----------------------------	----------------------------	-----------------

Abbildung 12: Wertebereiche der Schichten

Der Algorithmus erhält diese Bedingung als Invariante aufrecht, indem nach jedem Aktualisierungsschritt die Zugehörigkeit von Gitterzellen zu den Schichten überprüft und gegebenenfalls aktualisiert wird.

6.2.2 Algorithmus

Wie das Dense Field Verfahren und die Narrow Band Methode ist auch das Sparse Field Verfahren ein iterativer Algorithmus zum Lösen der Levelsetgleichung bis zur Konvergenz. Eine grobe Darstellung lautet:

Sparse Field Levelset Algorithmus

1. Konstruiere **active layer** (initiales Zero-Levelset)
2. Konstruiere innere und äussere **inactive layers**
3. Wiederhole bis zum Konvergenzkriterium:
 4. Berechne $\Delta\Phi$ für **active layer** Gitterzellen mittels eines numerischen Verfahrens zur Annäherung der Evolutionsgleichung
 5. Aktualisiere Gitterzellen, die im **active layer** verbleiben und **inactive layer**-Nachbarn die in den Bereich des **active layer** kommen
 6. Aktualisiere Layerzugehörigkeit der Gitterzellen
 7. Aktualisiere **inactive layers** mit schneller Distanzapproximation (+/- city block distance zum **active layer** Wert im Einheitsgitter)

In den weiteren Abschnitten wird nun auf Details einzelner Punkte und die mögliche Realisierung eingegangen.

6.2.3 Datenstrukturen

Gitterpunkte werden durch Objekte repräsentiert, ein *Element* stellt einen Gitterpunkt dar. Am einfachsten lassen sich die Elemente der Schichten in Listen verwalten. Zusätzlich werden drei Gitter²³, dem diskretisierten Problemraum entsprechend, benötigt. Ein Gitter wird benötigt, um den Schichtindex der Zelle zu speichern. Die dadurch entstehende Redundanz zu den Informationen der Elemente in den Schichtlisten ist nötig, um $O(1)$ Aufwand sowohl für das Überprüfen des Status einer Gitterzelle, das Finden von Nachbarn eines bestimmten Status und die Aufzählung aller Elemente der gleichen Schicht zu ermöglichen. Zusätzlich enthält ein weiteres Gitter die

²³ Einfach realisierbar durch mehrdimensionale Arraydatenstrukturen

Funktionswerte der Levelsetfunktion und ein Gitter für die Codierung von, für den Ablauf des Algorithmus wichtigen, Aktionen, wie zum Beispiel der geplante Wechsel einer Gitterzelle in eine andere Schicht.

Die Nutzung von drei der Größe des Problemraumes entsprechenden Gittern bedeutet einen erheblichen zusätzlichen Speicherbedarf. Während des Ablaufs des Algorithmus sind diese Gitter jedoch nur im Bereich der Schichten mit relevanten Daten besetzt. Der Gesamtspeicherbedarf ist im Vergleich zur tatsächlich genutzten Menge der Daten also tendenziell sehr gering. Deshalb wurde ein optimiertes Verfahren eingesetzt, das Speicher nur für tatsächlich genutzte Bereiche der Gitter belegt. Das Verfahren wird detaillierter im Kapitel „Implementation“ beschrieben.

6.2.4 Konstruktion der Schichten

Zur Initialisierung des Sparse Field Verfahrens ist es nötig, den active layer und die umgebenden inactive layer zu konstruieren. Wichtig ist, dass eine Methode zur Konstruktion des active layer eine geschlossene Kontur garantiert. Denkbar ist eine interaktive Methode, in der der Nutzer den Verlauf der Startkontur vorgibt. Oft sind automatische Verfahren jedoch wünschenswert.

Eine Möglichkeit dazu ist Ausführung des Fast Marching Verfahrens, um eine Startkontur zu ermitteln. Wird das Fast Marching Verfahren nach einem beliebigen Zeitschritt wegen des Erreichens einer Abbruchbedingung beendet, gibt es stets eine Menge Punkte, deren Status „Trial“ ist. Diese Menge bildet garantiert eine geschlossene Kontur. Active points sind dann genau die Punkte der „Trial“-Menge. Ist statt dessen bereits eine vorzeichenbehaftete Distanztransformation im Gitter um eine vorgegebene Kontur vorhanden, können die active points durch Extraktion der Gitterzellen mit einem Nulldurchgang der Distanztransformation ermittelt werden. Ein algorithmisches Schema dazu ist in [10] angegeben.

Nach der Konstruktion des active layer werden nun schrittweise die inactive layer aufgebaut. Dazu werden alle Gitterzellen in der Nachbarschaft der nächstinneren Schicht der neuen Schicht hinzugefügt. Ein Algorithmus dazu durchläuft also die Schicht, um die eine weitere Schicht herum konstruiert werden soll und fügt alle Nachbarpunkte hinzu die noch keiner Schicht zugeordnet sind. Damit entsteht wiederum

eine geschlossene Kontur um die Schicht, die eine neue Schicht bildet.

Die Anzahl der benötigten Schichten hängt von der Ordnung der räumlichen Differenzoperatoren ab. Werden maximal Operatoren zweiter Ordnung genutzt, sind also zwei innere und zwei äußere Schichten um den active layer nötig.

6.2.5 Aktualisierung des active layer

Um den active layer zu aktualisieren, wird zunächst die Änderung des Wertes der Levelsetfunktion $\Delta\Phi$ für jede Gitterzelle berechnet, die Teil des active layer ist. Dazu wird die numerische Näherung für die Levelset-Evolutionsgleichung genutzt. Für jedes Element des active layer wird dann geprüft, ob sich der Funktionswert nach der Aktualisierung innerhalb oder außerhalb des gültigen Intervalls befinden würde. Ist der Wert innerhalb des Intervalls, wird dieses Element (die Gitterzelle) aktualisiert.

Wenn der Wert außerhalb des gültigen Intervalls liegen würde, wird diese Gitterzelle vorerst nicht aktualisiert. Stattdessen wird anhand des Funktionswertes festgestellt, in welcher Richtung das Wertintervall des active set verlassen wird und damit, ob sich das Element in die innere oder in die äußere Nachbarschicht des active layer bewegt. Sei L die Zielschicht des Gitterpunkts, werden nun alle Nachbarn des Gitterpunkts, die in der gegenüberliegenden Zielschicht $-L$ liegen, betrachtet (also alle Nachbarerelemente in der Schicht „1“, wenn sich der active point in die Schicht „-1“ bewegt). Die Nachbarpunkte werden nach folgendem Schema aktualisiert:

$$\Phi_{\text{Nachbar}} = \Phi_{\text{Active}} + \Delta\Phi + \begin{cases} 1, & \text{wenn äußere Nachbarschicht} \\ -1, & \text{wenn innere Nachbarschicht} \end{cases}$$

Wurde ein Gitterelement in einer der Nachbarschichten in diesem Zeitschritt schon so aktualisiert (durch einen anderen active point), wird der nach der oben aufgeführten Formel neu ermittelte Wert mit dem bereits gesetzten Wert verglichen und geprüft, ob der neue Wert die Gitterzelle näher am Zero-Levelset platzieren würde. Für die innere Nachbarschicht wird deshalb der größere der beiden Werte und für die äußere Nachbarschicht der kleinere der beiden Werte ausgewählt.

Weil die Elemente in benachbarten Layern aufgrund der später ausgeführten Cityblock-Distanztransformation auf gleichbleibendem Abstand gehalten werden, führt die Bewegung des active point aus dem active layer dazu, dass die Nachbarn dieses

Elements in den active layer „gezogen“ werden. Um diese Bewegung später auszuführen, wird der active point gespeichert. Es werden dazu zwei Listen benutzt, die die Gitterpunkte speichern, die das active set in die nächste innere oder äußere Schicht verlassen.

Eine gegenläufige Bewegung zweier benachbarter active layer Zellen würde eine Lücke in der Kontur erzeugen. Deshalb wird ein Element nicht bewegt (und die Nachbarn nicht wie oben beschrieben aktualisiert), wenn festgestellt wird, dass ein Nachbarelement bereits für eine Bewegung in die andere Richtung eingeplant ist. Dazu kann im oben erwähnten Gitter zur Kodierung von Aktionen eine Markierung gesetzt werden, wenn ein Gitterpunkt bewegt wird. Vor dem Bewegen eines Gitterpunktes kann nun die Nachbarschaft nach active points abgesucht werden, für die eine entgegen gerichtete Bewegung bereits geplant ist.

Es ist wichtig, die tatsächlichen Aktualisierung der Levelset-Funktionswerte erst auszuführen, nachdem der Wert $\Delta\Phi$ aller Zellen des active layer bestimmt wurde. Andernfalls würden bereits aktualisierte Zellen das Ergebnis ihrer Nachbarzellen beeinflussen, weil die Levelsetgleichung über finite Differenzen angenähert wird. Statt die Elemente sofort zu aktualisieren (wie aus Verständnisgründen oben beschrieben) werden die nötigen Aktualisierungen tatsächlich erst vorgemerkt und am Ende des Aktualisierungsschritts gesammelt ausgeführt.

6.2.6 Aktualisierung der Schichtzugehörigkeit

Ändern sich die Werte von Gitterzellen, die Teil einer Schicht sind, so, dass sie außerhalb des jeweiligen Wertebereichs der Schicht liegen, muss die Gitterzelle einer anderen Schicht zugewiesen oder ganz aus den Schichten entfernt werden. Ebenso müssen Gitterzellen neu in die äußeren Schichten aufgenommen werden, wenn die Front diesen Bereich erreicht.

Am einfachsten lassen sich die Schichtwechsel mit zwei Listen verwalten: eine Liste für Elemente, die auswärts bewegt werden und die zweite für Elemente, die inwärts bewegt werden. Diese Listen werden in [13, 14] als *status lists* bezeichnet. Nach der Aktualisierung des active layer enthalten die Listen die Elemente, die sich in die erste innere beziehungsweise äußere Schicht bewegen. Wie bereits erwähnt, führt die

Bewegung eines Elements dazu, dass die in der zur Bewegungsrichtung entgegengesetzten Schicht liegenden Nachbarn des Elements praktisch „hinterhergezogen“ werden müssen. Diese Punkte werden ebenfalls in einer status list gesammelt. Das Ergebnis der Abarbeitung einer status list ist deshalb eine Ergebnisliste, die als status list im nächsten Schritt wiederrum abgearbeitet wird, bis die äußerste Schicht erreicht ist. Die Verarbeitung der status list einer Schicht erzeugt die status list für den nächsten Tausch.

Dabei wird die Schichtzugehörigkeit wie folgend dargestellt aktualisiert.

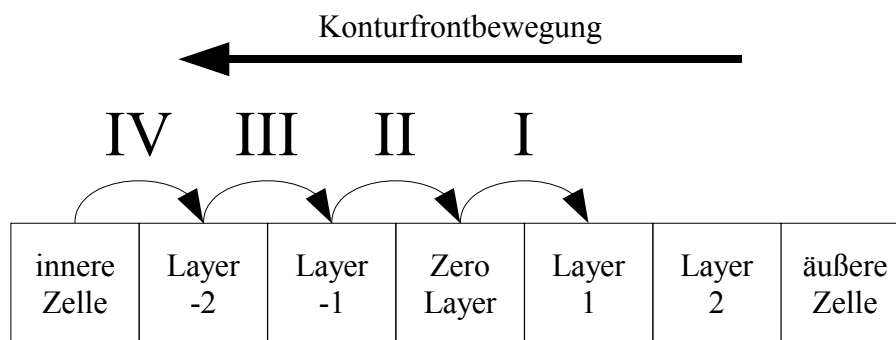


Abbildung 13: Ablauf des Schichtwechsels

Die Layerzugehörigkeit der Elementen wird also wie folgt aktualisiert:

1. Elemente des active layer aus dem active layer in eine der Nachbarschichten
2. Elemente der Nachbarschichten des active layer in den active layer
3. Elemente aus den inneren Schichten, die „nachrücken“ (in der Grafik nicht dargestellt, weil es keine inactive layer zwischen anderen inactive layers gibt)
4. Elemente der äußersten Schichten
5. Elemente außerhalb der Schichten, die in die beiden äußeren Schichten rücken (hier werden keine Elemente „nachgezogen“)

6.2.7 Aktualisierung der inactive layer

Nach der Abarbeitung der *status lists* sind alle Elemente in den korrekten Schichtenlisten, das heißt, den Gitterpunkten ist die korrekte Schicht zugewiesen. Die Werte der Elemente sind jedoch noch nicht korrekt. Deshalb werden nun die Gitterzellen der inactive layer mit der schnellen Distanztransformation aktualisiert. Damit werden die Levelsets in den äußeren Schichten wieder in einem gleichmäßigen Abstand um das Zero-Levelset angeordnet. Die Aktualisierung findet von den inneren Schichten zu den äußeren Schichten statt, die beiden äußersten Schichten werden also

zuletzt aktualisiert. Zum Aktualisieren eines Elements wird die Nachbarschaft (zum Beispiel 4-Nachbarschaft in einem zweidimensionalen Gitter) nach Elementen abgesucht, die eine Schicht weiter innen (näher dem Zero-Levelset) liegen. Von diesen Elementen wird das Element ausgewählt, das den geringsten Abstand zum Zero-Levelset hat. Hat man für das Äußere der Kontur positive Funktionswerte und für das innere der Kontur negative Funktionswerte definiert, wird also in den äußeren Schichten das Element mit dem kleinsten Wert und in den inneren Schichten das Element mit dem größten Wert ausgewählt. Zur Aktualisierung wird dann der Cityblock-Abstand (ist gleich 1) zu dem Wert des ausgewählten Nachbargitterpunkts addiert (äußere Schicht) oder von ihm subtrahiert (innere Schicht).

Wird für ein Element kein Nachbarelement in der nächstinneren Schicht gefunden, wird das Element der nächstäußeren Schicht zugewiesen. Auf diese Weise werden Elemente, die durch eine Bewegung eines Elements aus dem active layer weiter auswärts bewegt werden müssen, um „Platz zu machen“, in die äußeren Schichten bewegt. Befindet sich das Element bereits in einer der beiden äußersten Schichten, wird es stattdessen gelöscht.

7 Implementation

Der mit Abschluß dieser Arbeit vorliegende Prototyp wurde vollständig in der Programmiersprache Java erstellt. Die sich daraus ergebenden Nachteile in der Ausführungsgeschwindigkeit und dem generell höheren Speicherbedarf im Vergleich zu nativ kompilierten²⁴ Programmiersprachen wie C++ wurden im Hinblick auf die durch höhere Abstraktion einfachere Lesbarkeit, schnellere Entwicklung und die potenziell geringere Anfälligkeit für Programmierfehler in Kauf genommen. Auch konnten einige der Nachteile zumindest teilweise durch Optimierungen kompensiert werden, die im Verlauf dieses Kapitels beschrieben werden.

Der Quelltext gliedert sich in Pakete für die eigentlichen Algorithmen, Bildfilter, Programmsteuerung und die grafische Oberfläche. Das Programm ist als grafische Applikation mittels des Java Swing Frameworks für grafische Benutzeroberflächen gestaltet. Neben den eigentlichen Segmentierungsalgorithmen wurde ein Framework für die Benutzung von Bildfiltern und Schichtdatensätze implementiert, um das Programm einfach erweitern zu können. Langwierige Berechnungen werden in separaten Threads ausgeführt, um die Reaktionsfähigkeit der Nutzeroberfläche zu erhalten und eine asynchrone Visualisierung zu ermöglichen.

Während der Implementation wurden zur Funktionsprüfung Tests auf synthetischen Daten ausgeführt. Dabei handelte es sich um konstruierte Testfälle mit erwarteten Verhalten des Verfahrens. Zusätzlich wurden für alle verwandten Datenstrukturen während des Programmablaufs Konsistenz- und Plausibilitätsprüfungen durchgeführt, um Fehler zu finden.

7.1 Eingabedaten

Die Testdaten liegen in Dateien entsprechend dem DICOM Standard²⁵ vor. Dieser Standard definiert Regeln für den Austausch medizinischer Bilddaten. Das in Abschnitt 10 beschriebene Dateiformat besteht aus einem Header und den eigentlichen Bilddaten.

²⁴ Direkt in geeigneten Mikrocode für den Prozessor des Zielsystems übersetzt

²⁵ Digital Imaging and Communications in Medicine Standard, die komplette Spezifikation ist online verfügbar unter:
<http://medical.nema.org/>

Die enthaltenen Bilddaten können unkomprimiert, verlustfrei komprimiert aber auch verlustbehaftet komprimiert sein. Sie können auch in einer von den Tagdaten separaten Datei vorliegen.

Im Header sind Informationen über das Format der Bilddaten, aber auch Angaben über das Aufnahmeverfahren und Aufnahmezeit, das dazu verwendete Gerät und dessen Hersteller sowie persönliche Informationen über den Patienten enthalten. Die Speicherung dieser Informationen geschieht über die Aufführung von Schlüssel-Wert-Paaren. Die zulässigen Schlüssel sind im dritten Abschnitt der Beschreibung des DICOM-Standards definiert und werden als *Tags* bezeichnet. Für den Ablauf des Algorithmus wichtige Informationen sind über die Tags *PixelSpacing* und *SliceSpacing* definiert. Sie enthalten Informationen über den Abstand der Pixel in einer Schicht beziehungsweise den Abstand zwischen den Schichten. Sie sind nötig, wenn die Anisotropie der Voxel berücksichtigt werden soll.

Innerhalb des Prototyps werden die DICOM Dateien mit der quelloffenen, frei verfügbaren *dcm4che* Bibliothek²⁶ eingelesen. Die Anbindung erfolgt über die standardisierte *ImageIO* Schnittstelle der Java API Spezifikation. Gleichzeitig stellt *dcm4che* eine Schnittstelle zur Verfügung, um die oben erwähnten Metadaten zu extrahieren.

7.2 *Programmeffizienz*

7.2.1 Speicherbedarf

Die verwendeten Datensätze und Algorithmen, ohne besondere speichersparenden Maßnahmen umgesetzt, benötigen sehr viel Arbeitsspeicher. So war es nicht möglich, Tests mit einem kompletten CT-Datensatz auf einem System mit einem Gigabyte Hauptspeicher durchzuführen. Deshalb musste der Speicherbedarf reduziert werden. Die Fast Marching und Sparse Field Levelset Verfahren benutzen jeweils mehrere dreidimensionale Matrizen mit den Abmessungen der Eingabedatensätze. Für eine einfache Implementation bietet sich die Nutzung dreidimensionaler Arrays an. Ein großer Teil des insgesamt benötigten Speichers entfällt auf diese Arrays. Typischerweise

²⁶ <http://sourceforge.net/projects/dcm4che/>

wird jedoch nur ein kleiner Teil dieser Datenstrukturen tatsächlich benutzt – die Anforderung des gesamten Arrays verschwendet also einen großen Anteil des Speichers.

Aus diesem Grund wurde eine Ersatzdatenstruktur implementiert, die im typischen Anwendungsfall einen deutlich geringeren Speicherbedarf als ein entsprechendes Array hat. Grundgedanke dieser Datenstruktur ist es, annähernd nur für die genutzten Teile des Arrays tatsächlich Speicher zu belegen. Dieses als „Tiling“ (Kachelung) bekannte Verfahren wird in der Informatik häufig zum Sparen von Speicherplatz verwendet. Dazu werden die dreidimensionalen Arrays in kleinere Würfel (die „Kacheln“) unterteilt. Die Kacheln sind wiederum als Arrays realisiert. Sie sind aber im Vergleich zum ursprünglichen Array sehr klein.

Der Zugriff auf die Datenstruktur erfolgt wie bei Arrays über Indizes für jede einzelne Dimension. Aus den absoluten Indizes muss dann die entsprechende Kachel und die Position innerhalb der Kachel errechnet werden. Bei einem lesenden Zugriff auf einen nicht durch eine Kachel belegten Index wird zunächst ein definierbarer Standardwert, zum Beispiel Null für numerische Daten, zurückgegeben. Der Speicher für die entsprechende Kachel wird erst beim ersten Schreibzugriff belegt.

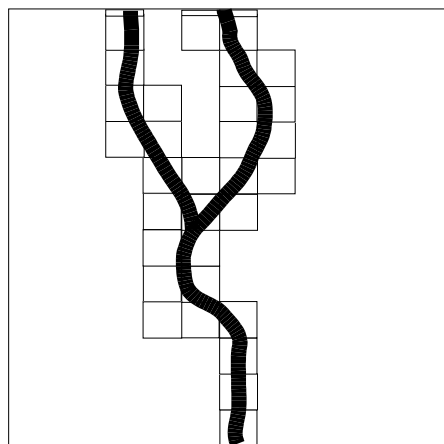


Abbildung 14: Kachelung eines zweidimensionalen Bereichs

Ein Nachteil dieses Verfahrens ist die im Vergleich zu einfachen Arrays geringere Ausführungsgeschwindigkeit, weil für die Umsetzung von absoluten Indizes zu tatsächlichen Adressen bei jedem Zugriff eine Ganzzahldivision und eine Modulooperation nötig sind. Weiterhin belegt das Verfahren zwar den Speicher verzögert, gibt ihn jedoch nicht wieder frei, wenn Kacheln nicht mehr benutzt werden –

also wieder komplett mit dem Standardwert belegt sind. Dieses Problem ließe sich nur mit einer in regelmäßigen Abständen wiederholten Prüfung des Inhalts aller Kacheln vermeiden. Das ist aber ein relativ aufwändiger Vorgang. Sollte die Datenstruktur nahezu vollständig besetzt werden, ist zudem der Speicherbedarf höher als bei der Nutzung einfacher Arrays. Für diesen Fall ist die Datenstruktur ungeeignet. Eine so starke Auslastung der Datenstruktur ist im Rahmen der Arbeit jedoch nicht zu erwarten, weil die Blutgefäße nur einen geringen räumlichen Anteil des in den CT-Daten erfassten Körperbereichs belegen.

7.2.2 Laufzeit

Selbst effiziente Levelsetverfahren, wie der Sparse-Field-Algorithmus sind sehr aufwändig in der Berechnung. Zusätzlich ergibt sich, wie erwähnt, durch die Verwendung der Programmiersprache Java ein Nachteil der Ausführungsgeschwindigkeit im Vergleich zu nativ kompilierten Sprachen wie C++. Deshalb wurden einige Maßnahmen zum Verringern der benötigten Ausführungszeit getroffen.

Einen großen Einfluß auf die benötigte Laufzeit des Programms hat das häufig nötige Erzeugen und Freigeben von Objekten – zum Beispiel von Elementen für die Schichten im Sparse Field Algorithmus. Erzeugung und Bereinigung von Objekten durch das automatische Speicherverwaltungssystem (garbage collection) sind innerhalb der Java-Laufzeitumgebung aufwändige Prozesse. Das massive Erzeugen von Objekten für sich oft wiederholende, kurzfristige Aufgaben sollte deshalb vermieden werden. Dazu wurde ein Zwischenspeicher implementiert, der nicht mehr benötigte Objekte aufnimmt und auf Anforderung zum Wiederverwenden abgibt. Ist der Zwischenspeicher bei einer Anforderung leer, wird ein neues Objekt erzeugt und zurückgeliefert. Wenn der Zwischenspeicher einen definierbaren Maximalbestand erreicht hat, werden zurückgegebene Objekte zur Löschung durch die Laufzeitumgebung freigegeben.

Weiterhin wurde für oft benötigte Datenstrukturen der Designgrundsatz der höchstmöglichen Lokalität bewußt umgangen. Dieser Grundsatz der objektorientierten Programmierung besagt, dass Daten im jeweils kleinstmöglichen Sichtbarkeits- / Lebenszykluskontext gehalten werden sollten (zur Vermeidung von *scope pollution*²⁷).

²⁷ Die „Verschmutzung“ eines Variablensichtbarkeitskontexts durch Variablen eines tieferen Kontexts.

Viele Datenstrukturen werden nur innerhalb des Ablaufs einer Methode genutzt – sollten deshalb also lokal innerhalb der Methode bei jedem Aufruf erzeugt werden. Bei einer häufigen Nutzung der Methode kommt es dabei aber zu einer häufigen Erzeugung und Freigabe dieser Datenstrukturen, was, wie bereits aufgeführt, für komplexe Objekte sehr aufwändig ist. Solche komplexen Datenstrukturen wurden deshalb zur Wiederverwendung in einem höheren Kontext als nötig (als Instanzvariablen) deklariert. Oft wurde außerdem eine weitere Standardmethode der Informatik – die bis zur tatsächlichen Nutzung verzögerte Auswertung von eventuell nicht benötigten Ausdrücken²⁸ – angewendet.

Eine deutliche Verbesserungen der Verarbeitungsgeschwindigkeit könnte der Einsatz eines in Java vergleichsweise einfach implementierbaren Multithreading-Verfahrens²⁹ in einer Mehrprozessorumgebung bieten. Das Sparse-Field Levelsetverfahren ist für die parallele Verarbeitung durch mehrere Prozessoren sehr gut geeignet, weil die Verarbeitung der Schichtelemente in einer beliebigen, unabhängigen Reihenfolge, also auch verteilt, erfolgen kann und die tatsächlichen Aktualisierungen erst am Ende der Berechnungen gesammelt ausgeführt werden. Zudem nimmt es den weitaus größten Anteil am gesamten Rechenzeitaufwand der implementierten Segmentierung ein. Diese Erweiterung wurde im Rahmen dieser Arbeit nicht implementiert.

7.3 Nutzerschnittstelle

Das Augenmerk der Entwicklung lag auf den Segmentierungsalgorithmen. Deshalb existiert nur ein Prototyp für die grafische Nutzeroberfläche. Die meisten Optionen der Algorithmen sind nicht über die Oberfläche, sondern über Konstanten im Programmquelltext konfigurierbar.

Der Prototyp kann sowohl DICOM Bilder als auch einfache, zweidimensionale Bilder (zum Beispiel in den Formaten Bitmap und JPEG) einlesen. Dies geschieht über den Menüpunkt „File – Load Image“. Wird ein aus mehreren Schichten bestehendes Bild geladen, kann mit der „Bild rauf“ / „Bild runter“ Taste durch die einzelnen Bilder

Das erhöht die Gefahr einer unbeabsichtigten, unbemerkten Überlagerung einer Variable durch eine andere Variable gleichen Namens die zu Programmfehlern führen kann.

28 Auch als lazy evaluation beziehungsweise lazy initialization bezeichnet

29 Aufspaltung des Programms in mehrere Kontrollflüsse, die quasi gleichzeitig die selbe Datenbasis bearbeiten

geblättert werden.

Das Setzen von Startmarkierungen ist mit der linken Maustaste möglich. Die Markierung wird stets in der aktuell angezeigten Schicht des Datensatzes gesetzt. Gesetzte Markierungen werden durch kleine rote Kreuze angezeigt. Mit der rechten Maustaste wird die zuletzt gesetzte Markierung entfernt. Eine wiederholte Betätigung der rechten Maustaste löscht also die Startmarkierungen in umgekehrter Reihenfolge ihrer Erzeugung.



Abbildung 15: Screenshot

Über den Menüpunkt „Action – Segment“ wird das Segmentierungsverfahren gestartet. Fortschrittmeldungen werden über die Java Systemkonsole ausgegeben. Während der Berechnung ist ein Blättern innerhalb der Daten durch die Benutzung eines eigenen Kontrollflusses (Thread) für das Segmentierungsverfahren möglich. So können die während des Ablaufs erzeugten Visualisierungen der Front auf einer beliebigen Schicht beobachtet werden.

8 Evaluation

Ziel der Evaluation ist die Bewertung der Qualität der Ergebnisse des implementierten Verfahrens. Sie soll einen praktischen Einsatz mit einem begründeten Urteil rechtfertigen oder aber verwerfen. Die Anwendung im medizinischen Bereich erfordert eine besonders kritische Prüfung, weil unter Umständen Entscheidungen über die Durchführung von Operationen anhand der gelieferten Daten gefällt werden sollen. Ein



Abbildung 16:
Segmentierte
Verzweigung (innere
Jugularvene)

weiteres Ziel der Evaluation ist es, Aussagen über besonders geeignete Einsatzbedingungen und Einschränkungen in der Anwendung zu treffen. Typisch ist die Nutzung von statistischen Methoden, um die Ergebnisse der Tests zu verdichten. Die Anzahl der Tests sollte deshalb repräsentative Menge entsprechen und gewährleisten, dass Ausreißer das Gesamtergebnis nicht wesentlich verfälschen können.

Das entscheidende Kriterium in dieser Arbeit ist die Genauigkeit der Segmentierung. Um die Genauigkeit zu beurteilen, werden die Ergebnisse mit Referenzsegmentierungen bekannter Güte verglichen. Die zur Verfügung stehenden Referenzsegmentierungen einzelner Gefäße wurden am Institut für Simulation und Grafik der Universität Magdeburg mittels einer handkorrigierten Watershed-Transformation erstellt.

Ein weiteres Qualitätskriterium ist die Abhängigkeit des Verfahrens von der Parameterisierung. Von Interesse ist, wie sensibel der Algorithmus auf Änderungen einzelner interner (Konstanten im Algorithmus) oder externer (Variierung der Nutzervorgaben, wie zum Beispiel Saatpunkten) Parameter reagiert. Eine statistisch relevante Menge an Tests ist dazu erforderlich. Sehr wichtig ist auch, dass bei der Anwendung des Verfahrens mit den selben internen Parametern auf der gesamten Menge der Testdaten akzeptable Ergebnisse erreicht werden können. Das verringert die Komplexität der Nutzerinteraktion, weil sie nicht zur Anpassung durch den Nutzer sichtbar gemacht werden müssen. Ausserdem ist die Funktion und Wirkung interner Parameter, wie zum Beispiel Konvergenzgrenzen, für den Nutzer häufig nur

schwer zu erfassen.

Schließlich wird eine vergleichende Segmentierung auf MRT-Datensätzen vorgenommen, die für die geforderte Aufgabe theoretisch besser geeignet sind, weil sich weiche Gewebe in MRT-Aufnahmen deutlicher voneinander abgrenzen als in CT-Aufnahmen. Hier soll geprüft werden, ob dieser Vorteil durch das implementierte Verfahren ausgenutzt werden kann.

8.1 Vergleichsverfahren

Zur Einschätzung der Segmentierungsgüte des Verfahrens wurden die Ergebnisse mit den oben erwähnten Referenzdaten verglichen. Sie liegen als Binärbilder, ebenfalls im DICOM Format, vor. Referenzsegmentierungen existieren für die Halsschlagader (*aorta carotis communis*) und die innere³⁰ Jugularvene (*vena jugularis interna / externa*) – die größten Gefäße im Bereich der Datensätze. Die Güte dieser Referenzdaten ist nicht als Goldstandard für eine anatomisch korrekte Segmentierung anzusehen, da das verwendete Verfahren selbst nicht optimal ist. Zumindest ermöglicht es jedoch einen Vergleich der Verfahren untereinander.

Nach dem Abschluss der Sparse-Field-Levelset Phase wird das Ergebnis durch das Programm schicht- / pixelweise mit den gegebenen Referenzdaten verglichen. Dabei werden drei Fälle betrachtet:

- Übereinstimmung: Durch das Levelset segmentiert und in der Referenz segmentiert
- Untersegmentierung: In der Referenz segmentiert jedoch nicht durch das Levelset segmentiert
- Übersegmentierung: Durch das Levelset segmentiert jedoch nicht in der Referenz segmentiert

Alle Pixel, die weder im Referenzdatensatz noch durch das Levelset segmentiert wurden, werden nicht betrachtet. Die aufgeführten Daten werden pro Schicht erfasst. Zusätzlich werden Statistiken, wie die mittlere Übereinstimmung über alle Schichten, und Extremwerte, wie minimale und maximale Übereinstimmung, ermittelt. Außerdem wurde ein Sichtvergleich der überlagerten Vergleichssegmentierungen mit der Referenz

³⁰ Segmentierungen für die äußere Jugularvene stand in einem Datensatz zur Verfügung

ausgeführt, um „Auslauf“-Effekte zu erkennen.

Die internen Parameter des Verfahrens wurden während der Tests nicht verändert, obwohl so bessere Segmentierungsergebnisse zu erzielen gewesen wären. Der Grund dafür ist, dass Funktion und Zusammenwirken der einzelnen Parameter für den Nutzer nicht einfach verständlich sind. Deshalb muss das Ziel sein, die Parametrisierung vor dem Benutzer so weit wie möglich zu verbergen.

8.2 Ergebnisse

Es standen verschiedene CT-Datensätze zum Testen des Verfahrens zur Verfügung. Die Auflösung innerhalb der Schichten beträgt 512x512 Pixel. Die Datensätze haben zwischen 37 und 262 Schichten. Dementsprechend unterscheiden sich die Aufnahmen wesentlich im Abstand der Pixel in der Schicht und dem Abstand zwischen den Schichten und der Schichtdicke. Es wurden insgesamt 29 Gefäße in sieben CT-Datensätzen segmentiert.

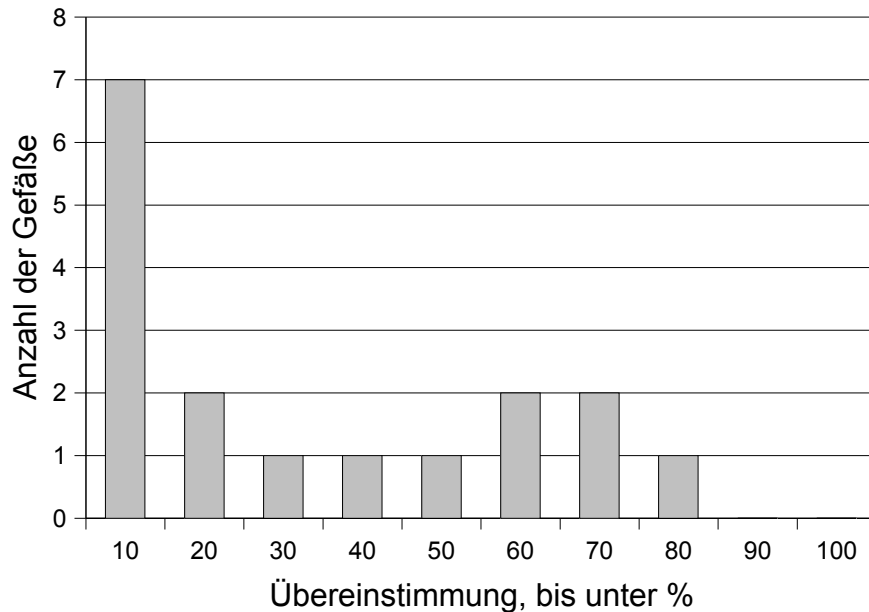
Die Segmentierungsergebnisse sind relativ schlecht. Der hauptsächliche Grund ist, dass die Kontur während der Segmentierung in das umgebende Gebiet ausgelaufen ist. Ein weiterer Grund ist, dass das Verfahren, wie beabsichtigt, Verzweigungen der Gefäße mitsegmentiert hat, in den Referenzdaten jedoch nur die erwähnten Hauptgefäße segmentiert wurden. Das führte zu berechneten Übersegmentierungen, obwohl das Gefäß korrekt segmentiert wurde. Schliesslich war auch noch eine kontinuierliche, ringförmige Untersegmentierung in Bereichen in der die Levelsetkontur der Referenzkontur gefolgt ist, zu erkennen. Die Segmentierung durch das Levelsetverfahren war konservativer konfiguriert, um ein Auslaufen zu verhindern. Diese drei Gründe führten zu schlechten Übereinstimmungsquoten mit der Referenzsegmentierung.

Der überwiegende Teil der Blutgefäße wurde mit einer Übereinstimmungsquote von unter fünfzig Prozent segmentiert, mehr als die Hälfte davon aufgrund eines Auslaufens der Kontur gar nicht³¹. Ein Auslaufen fand, wenn, dann aber meistens bereits in der Fast Marching Phase statt. Das Ergebnis war deshalb besonders dann gut, wenn das

³¹ Diese Gefäße sind im Diagramm mit einer Übereinstimmungsgüte von unter zehn Prozent aufgeführt

Blutgefäß auf dem gesamten Verlauf gut zur Umgebung abgegrenzt war und sich nicht an andere Gefäße angelagert hat. Für zwei der Datensätze – acht Gefäße – waren keine Referenzdaten verfügbar. Sie sind im nachfolgenden Diagramm nicht aufgeführt. Diese Gefäße wurden, nach Einschätzung durch eine Sichtprüfung, etwa zur Hälfte ohne Auslaufen der Kontur segmentiert.

Vergleichsegmentierung



Aufgrund der geringen Menge der zur Verfügung stehenden Testdatensätze konnten keine Rückschlüsse gezogen werden, ob die Testergebnisse sich bei Datensätzen mit geringerem Schichtabstand, und damit potenziell besser verfolgbaren Gefäßen, verbessern. Eine Sensibilitätsanalyse der internen Parameter konnte nicht mehr durchgeführt werden.

Weil Magnetresonanztomografie weiche Gewebetypen besser unterscheiden kann, wurde zum Vergleich eine Segmentierung auf einem MRT-Datensatz ausgeführt. Der zur Verfügung stehende Datensatz war jedoch sehr grob aufgelöst und kontrastarm, so dass die Kontur auslief und die Segmentierung nicht erfolgreich war.

Die ausführlichen Testergebnisse sind auf der beigelegten CD im PDF-Format verfügbar.

9 Zusammenfassung und Ausblick

9.1 Erreichtes

Mit dem Abschluss dieser Arbeit ist ein anpassungsfähiges, erweiterbares Segmentierungsverfahren für dreidimensionale Bilddaten basierend auf impliziten aktiven Konturen implementiert. Die beiden zugrunde liegenden Verfahren – der Fast Marching sowie der Sparse Field Levelset Algorithmus – wurden sehr effizient in Bezug auf Speicher- und Rechenzeitverbrauch umgesetzt.

Das Verfahren wurde auf mehreren CT- und einem MRT-Datensatz getestet und mit einer Referenzsegmentierung verglichen. Dabei zeigte sich, dass das Verfahren eine manuelle Expertensegmentierung nicht ersetzen, jedoch unterstützen kann, indem eine grundlegende Basissegmentierung erreicht wird. Verzweigungen der Blutgefäße werden generell durch das Verfahren sehr gut erkannt und verfolgt. Komplizierte Topologien, wie zum Beispiel das gelegentlich auftretende Wiederverschmelzen von Blutgefäßen nach einer bereits erfolgten Aufteilung können gut modelliert werden. Insbesondere die erhoffte Segmentierung besonders feiner Strukturen konnte jedoch nicht erreicht werden. Zudem ist das Verfahren nicht sehr robust gegenüber dem Auslaufen von Konturen.

9.2 Probleme

9.2.1 Auslaufen der Kontur

Das Auslaufen der Kontur ist weiterhin ein großes Problem, auch wenn es im Vergleich zu einem einfachen Regiongrowing Verfahren reduziert werden konnte. Besonders während des Fast Marching Verfahren trat oft ein Auslaufen während der Tests auf. Das eher seltene Auslaufen der Levelsetkontur tritt dann auf, wenn die Blutgefäße großflächig an andere Gewebe mit einer ähnlichen Grauwertdarstellung im CT-Bild angelagert sind. Das Auslaufen kann hier nicht durch das Verhindern einer hohen lokalen konvexen Krümmung gestoppt werden, weil ein breiter Bereich der Front die eigentliche Gefäßgrenze verlässt.

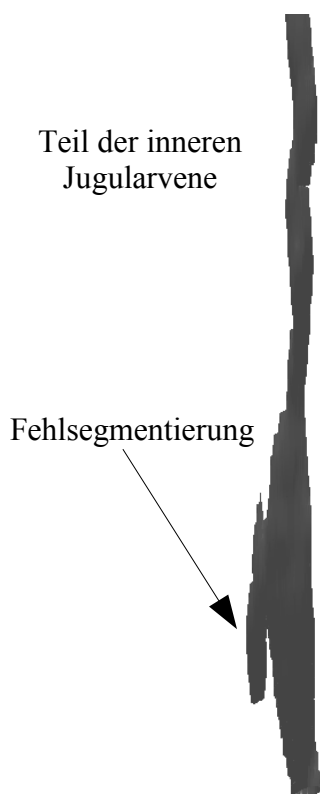


Abbildung 17:
Segmentierungsfehler (Auslaufen
der Kontur)

Ein vielversprechender Ansatz ist das in [12] beschriebene Verfahren der Zentralachsenberechnung für die Blutgefäße vor der levelsetbasierten Segmentierung. Eine Analyse des Verlaufs der Achse durch die Schichten könnte genutzt werden, um ein Auslaufen in der Fast Marching Phase zu erkennen. Ein Hinweis darauf wäre ein plötzlicher Sprung im Verlauf der Achse. Die Evolution der Levelsetkontur wird in [12] zusätzlich durch einen Term, der den Abstand zur Zentralachse in der Aufnahmeebene berücksichtigt, kontrolliert. Es wird so eine kreisförmige Kontur in der Ebene unterstützt. Allerdings wird erwähnt, dass das Verfahren Schwächen im Bereich von Verzweigungen hat, weil hier kein kreisförmiger Gefäßquerschnitt mehr existiert.

9.2.2 Feine Strukturen

Ein Problem ist, dass sehr feine Strukturen aufgrund ihrer hohen lokalen Krümmung nicht segmentiert werden – die regularisierende Kraft verhindert dies. Um dieses Problem zu umgehen, müsste der Krümmungsterm in Regionen mit sehr kleinen Konturen abgeschwächt oder ganz deaktiviert werden. Das erfordert jedoch eine Kohärenzanalyse der Konturgitterpunkte. So könnte festgestellt werden, wenn ein Gitterpunkt Teil einer sehr kleinen Kontur in der Ebene ist und unter dem Krümmungsterm kollabieren würde. Diese Informationen müssen nach jedem Schritt zumindest aktualisiert oder sogar komplett neu berechnet werden. Das ist sehr aufwändig, weil implizite Konturen gerade die Verwaltung einer konkreten Repräsentation vermeiden wollen. Ein Hybridmodell wie in [21] genutzt, könnte diesen Schritt vereinfachen.

9.3 Ausblick

Aufbauend auf dem Prototypen der Nutzerschnittstelle ist es wichtig, die Interaktionsfähigkeiten des Nutzers zu erhöhen. Gerade im medizinischen Bereich sind die Überwachungs- aber auch Eingriffsmöglichkeiten durch einen Experten wichtig, weil unter Umständen viel Verantwortung für die Genauigkeit übernommen werden muss. So könnte der Benutzer zum Beispiel die durch das Fast Marching Verfahren erzeugte Initialkontur inspizieren und korrigieren, bevor die levelsetbasierte Phase gestartet wird, da, wie erwähnt, die meisten Fehlsegmentierungen bereits vor der Levelsetphase entstehen. Denkbar wäre auch die Möglichkeit, gesperrte Bereiche zu definieren, in denen der Nutzer durch eine vorherige Sichtprüfung erkannt hat, dass das zu segmentierende Blutgefäß in einem Bereich sehr schwer von der Umgebung zu differenzieren ist und es zu einem Auslaufen der Kontur kommen könnte.

Ein weiterer Ansatzpunkt ist die Integration globaler Metriken. Ein während dieser Arbeit untersuchter Ansatz war die Beobachtung des Krümmungsprofils der Schichten während der Levelsetphase. Ein sich schnell änderndes Profil könnte ein Hinweis auf ein Auslaufen der Kontur sein. Dieser Ansatz wurde nicht umgesetzt, da dazu nach jeder Iteration des Levelsetverfahrens eine aufwändige Kohärenzanalyse der beteiligten Gitterpunkte nötig gewesen wäre. Hier ist ein schnelles Verfahren nötig, das die Kohärenz der Elemente ermittelt und aktualisiert.

Das Fast Marching Verfahren ist in der gegenwärtigen Implementation nicht geeignet, um die Initialkontur für die Levelsetphase zu liefern, weil es nicht sehr robust ist. Möglich wäre statt dessen zum Beispiel eine Kombination mit der für die Referenzdaten genutzten interaktiven, markerbasierten Watershed-Transformation. Diese Segmentierungen sind zwar relativ genau, die Konturen jedoch oft sehr unregelmäßig. Sie könnten als Initialkontur für das Levelsetverfahren genutzt werden.

Potenzial, die Segmentierungsergebnisse dieser Arbeit durch weiterführende Entwicklungen zu verbessern, ist also vorhanden.

10 Quellen

- [1] H. Lippert, R. Pabst – *Arterial Variations in Man*; J.F. Bergmann Verlag München, 1985
- [2] K. Fleischauer – *Benninghoff: Makroskopische und mikroskopische Anatomie des Menschen, 2. Band: Kreislauf und Eingeweide*; Urban und Schwarzenberg München – Wien – Baltimore, 13./14. Auflage, 1985,
- [3] J. A. Sethian – *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Materials Science*; Cambridge University Press, 1996
- [4] R. Malladi, J. A. Sethian – *A Realtime Algorithm for Medical Shape Recovery*; International Conference on Computer Vision, 1998
<http://math.lbl.gov/~malladi/Papers/iccv.ps.Z>
- [5] J. A. Baerentzen – *On the implementation of Fast Marching Methods for 3-D lattices*; Technical Report IMM-REP-2001-13, Technical University of Denmark Lungby, 2001
http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=841
- [6] L. Yatziv, A. Bartesaghi, G. Sapiro – *$O(N)$ Implementation of the Fast Marching Algorithm*; 2005
http://mountains.ece.umn.edu/~liron/fastmarching/O_N_fast_marching.pdf
- [7] J. A. Sethian – *Level Set Methods: An Act of Violence*; American Scientist, 1997
- [8] J. A. Sethian – *Fast Marching Methods*; SIAM Review Volume 41, Issue 2, 1999
- [9] J. A. Sethian, D. Adalsteinsson – *A Fast Level Set Method for Propagating Interfaces*; Journal of Computational Physics, Volume 118, Issue 2, 1995

- [10] R. Malladi, J. A. Sethian, B. C. Vemuri – *Shape Modelling with Front Propagation: A Levelset Approach*; IEEE Transactions on Pattern Recognition and Machine Intelligence, Volume 17, Issue 2, 1995
- [11] S. Osher, J.A. Sethian – *Front Propagating with Curvature Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations*; Journal of Computational Physics, Volume 79, 1998
<http://math.berkeley.edu/~sethian/Papers/sethian.osher.88.pdf.gz>
- [12] C. M. van Bommel, L. J. Spreeuwiers, M. A. Viergever, W. J. Niessen – *A Level-Set-Based Artery-Vein Separation in Blood-Pool Agent CR-MR Angiograms*; IEEE Transactions on Medical Imaging, Volume 22, Issue 10, 2003
- [13] R. T. Whitaker – *Isosurfaces und Level-Set Surface Models*; Technical Report UUCS-02-010 Utah School of Computing, 2002
<http://www.cs.utah.edu/techreports/2002/pdf/UUCS-02-010.pdf>
- [14] R. T. Whitaker – *A Level-Set Approach to 3D Reconstruction From Range Data*; The International Journal of Computer Vision, Issue 29, 1998
http://www.cs.utah.edu/~whitaker/sceneRecon/papers/sparse_report.pdf
- [15] J. A. Sethian, A. M. Popovici – *3D travelttime computation using the fast marching method*; Geophysics Volume 64, 2002
<http://math.berkeley.edu/~sethian/Papers/sethian.seismic.pdf.gz>
- [16] J. Rickett, S. Fomel – *A second-order fast marching eikonal solver*; Stanford Exploration Project Report, 2002
http://sepwww.stanford.edu/oldsep/sergey/sepsergey/fmsec/paper_html/
- [17] E. Rouy, A. Tourin – *A Viscosity Solutions Approach for Shape-From-Shading*; SIAM Journal on Numerical Analysis Volume 29 Issue 3, 1992

- [18] Y. Jin, A. Laine, C. Imielinska – *An Adaptive Speed Term Based on Homogeneity for Level-Set based Segmentation*; Columbia University New York, 2004
<http://medimage.bme.columbia.edu/images/publications/13-pdf.pdf>
- [19] A. L. N. Wong, H. Liu, P. Shi – *Segmentation of Myocardium Using Velocity Field Constrained Front Propagation*; Hongkong University of Science and Technology, 2002
<http://www.ee.ust.hk/~eeship/Papers/WACV02.pdf>
- [20] M. E. Leventon, O. Faugeras, W. E. L. Grimson, W.M. Wells III – *Level Set Based Segmentation with Intensity and Curvature Priors*; Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis, 2000
<http://splweb.bwh.harvard.edu:8000/pages/papers/leventon/mmbia00/mmbia00.pdf>
- [21] D. Magee, A. Bulpitt, E. Berry – *Combining 3D Deformable Models and Level Set Methods for the Segmentation of Abdominal Aortic Aneurysms*; British Machine Vision Conference, 2001
http://www.bmva.ac.uk/bmvc/2001/papers/27/accepted_27.pdf
- [22] L. Cheng, S. Osher – *Level set based Eulerian methods for multivalued traveltimes in both isotropic and anisotropic media*; Society of Exploration Geophysicists Technical Program Expanded Abstracts, 2003
<http://www.levelset.com/download/osher.pdf>
- [23] T. Deschamps – *Curve and shape extraction with minimal path and level-sets techniques. Applications to 3D medical imaging*; PhD Dissertation, Universität Paris, 2001
- [24] Q. Lin – *Enhancement, Extraction, and Visualization of 3D Volume Data*; PhD Dissertation, Universität Linköping April, 2003
<http://www.isy.liu.se/~qingfen/thesis/phdweb.pdf>

- [25] T. Deschamps, L. D. Cohen – *Fast Extraction of Tubular and Tree 3D Surfaces with front propagation methods*; Proceedings of the 6th International Conference on Pattern Recognition, 2002
http://math.lbl.gov/~deschamp/ps/deschamps_icpr02.pdf
- [26] T. Deschamps, L. D. Cohen – *Fast extraction of minimal paths in 3D images and applications to virtual endoscopy*; Medical Image Analysis 1, 2001
http://math.lbl.gov/~deschamp/ps/deschamps_media01.pdf
- [27] B. Jähne – *Digitale Bildverarbeitung*; Springer Verlag Berlin, 6. Auflage, 2005
- [28] K. D. Tönnies – *Grundlagen der Bildverarbeitung*; Pearson Studium München, 2005
- [29] M. Kass, A. Witkin, D. Terzopoulos – *Snakes: Active contour models.*; International Journal of Computer Vision, Issue 1, Number 4, 1987
- [30] S. M. Smith, J. M. Brady – *SUSAN - a new approach to low level image processing*; International Journal of Computer Vision, Volume 23 Issue 1, 1997.
- [31] T. McInerney, D. Terzopoulos – *Deformable Models in Medical Image Analysis: A Survey*; Medical Image Analysis Issue 1, 1996
<http://mrl.nyu.edu/~dt/papers/mia96/mia96.pdf>
- [32] J. A. Sethian, J. D. Strain – *Crystal Growth and Dendritic Solidification*; Journal of Computational Physics Volume 98, 1992.
- [33] C. Kirbas, F. K. H. Quek – *A Review of Vessel Extraction Techniques and Algorithms*; Wright State University Dayton, 2002
<http://vislab.cs.vt.edu/review/extraction.html>
- [34] S. Wesarg, E. A. Firlre – *Segmentation of Vessels: The Corkscrew Algorithm*; Proceedings of the SPIE Conference on Medical Imaging, 2004
http://a7www.igd.fhg.de/publications/swesarg/SPIE2004_Vessel_FP_C.pdf

[35] F. K. H. Quek, F. Kirbas – *Vessel Extraction in Medical Images by Wave-Propagation and Traceback*; IEEE Transactions on Medical Imaging, Vol.. 20, No. 2 , 2001

[36] T. McInerney, D. Terzopoulos – *T-snakes: topology adaptive snakes*; Medical Image Analysis Volume 4, 2000

10.1 Links

Computertomografie:

http://www.m-ww.de/enzyklopaedie/diagnosen_therapien/computertomographie.html

<http://www.iap.uni-bonn.de/P2K/tomography/>

http://www.prometheus.uni-tuebingen.de/player/dokument.jsp?_document=34&path=Seite01.htm

Angiografie:

<http://www.netdokter.de/ratschlaege/untersuchungen/angiographie.htm>

CFL-Bedingung:

<http://ourworld.compuserve.com/homepages/anima/cfl.htm>

Anatomie:

<http://isgwww.cs.uni-magdeburg.de/cv/lehre/MedVisualization/glossar.html>

<http://education.yahoo.com/reference/gray/>

Einführung zum DICOM-Standard:

<http://www.rsna.org/practice/dicom/intro/index.html>

11 Inhaltsverzeichnis der CD

Ordner src – Quelltext der Anwendung

Ordner javadoc – Automatisch generierte Dokumentation der Programmquellen

thesis.pdf – Diese Arbeit im pdf-Format

experiments.pdf – Detaillierergebnisse der Tests

12 Eigenständigkeitserklärung

Die vorliegende Arbeit wurde selbstständig von mir, nur unter Zuhilfenahme der angegebenen Quellen, erstellt.

Arne-Michael Törsel

Magdeburg, 31.08.2005