PATRICK SAALFELD

METHODEN ZUR SKIZZIERUNG VON GEFÄSSEN UND GEFÄSSERKRANKUNGEN MIT INTEGRIERTEM BLUTFLUSS

METHODEN ZUR SKIZZIERUNG VON GEFÄSSEN UND GEFÄSSERKRANKUNGEN MIT INTEGRIERTEM BLUTFLUSS

Masterarbeit
im Studiengang Informatik

Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Institut für Simulation und Graphik

EINGEREICHT VON: Patrick Saalfeld
MATRIKELNUMMER: 182300

BETREUER: Prof. Dr.-Ing. Bernhard Preim
Dipl.-Math. Kai Lawonn

GUTACHTER: Prof. Dr.-Ing. Bernhard Preim
Prof. Dr.-Ing. Andreas Nürnberger

EINGEREICHT AM: 11. März

ZUSAMMENFASSUNG

Diese Arbeit beschäftigt sich mit konkreten Methoden zum Skizzieren von Gefäßen, deren Erkrankungen und Behandlungsmöglichkeiten bei gleichzeitiger Simulation und integrierter Darstellung des Blutflusses. Die Methoden sollen zur Unterstützung des Arzt-Patient-Dialogs in der Patientenaufklärung genutzt werden. Hierzu werden Grundlagen aus den Bereichen der Medizin, der Strömungslehre und den skizzenbasierten Benutzungsschnittstellen vorgestellt. Für die Benutzungsschnittstellen werden zusätzlich Kriterien untersucht, die eine ergonomische Verwendung ermöglichen. Darauf folgt die konzeptionelle Beschreibung und anschließende Entwicklung eines Prototyps, welcher das Skizzieren der Gefäßstrukturen und Behandlungsmöglichkeiten erlaubt und dabei eine echtzeitfähige Simulation des Blutflusses ermöglicht. Die Entwicklung des Prototyps folgt dabei ergonomischen Gesichtspunkten für die Bedienung. Zur Bewertung des Prototyps werden qualitative und quantitative Evaluierungsmethoden angewendet.

INHALTSVERZEICHNIS

AKRONYME

CFD Computational Fluid Dynamics

CPU Central Processing Unit

DLIC Dynamic Line Integral Convolution

FBO Framebuffer Object

FDM Finite-Differenzen-Methode

FPS Frames pro Sekunde

GPGPU General Purpose Computation on Graphics Processing Unit

GPU Graphics Processing Unit

GUI Graphical User Interface

HCI Human Computer Interaction

LIC Line Integral Convolution

NPR nicht-photorealistisches Rendering

RBI Reality-Based Interaction

RGB Rot, Grün, Blau

SBI Sketch-based Interface

SBIM Sketch-based Interface for modeling

SPH Smoothed Particle Hydrodynamics

UI User Interface

WIMP Windows, Icon, Menu, Pointer

1

EINLEITUNG

„Menschen sind fortgeschrittene Problemlöser“ - das Attribut *fortgeschritten* erhalten wir durch die Fähigkeiten zu Be- und Ergründen, kreativ zu denken und durch Verstehen. Ein Mittel, welches uns allein oder in der Gruppe beim Lösen von Problemen und dem Kommunizieren unterstützt, sind Skizzen [53]. Daher werden in vielen Fachgebieten wie der Medizin Skizzen von z. B. Allgemeinmedizinern, Radiologen und Chirurgen angefertigt. Diese dienen nicht nur als Diskussionsgrundlage untereinander oder der Visualisierung eigener Gedanken, sondern auch als Kommunikationswerkzeug im Dialog mit dem Patienten. Mithilfe von Skizzen können dem Patienten bildhaft Elemente der Prävention, Diagnose, Therapie und Nachsorge illustriert werden.

Ein Teilgebiet der Medizin beschäftigt sich mit Erkrankungen der zentralen Versorgung des Menschen durch das Herz und Gefäße und nimmt eine wichtige Stellung in der Medizin und Volkswirtschaft ein. Diese Bedeutung resultiert aus der hohen Anzahl von Krankheits- und Todesursachen in den Industrieländern, die auf dieses Gebiet zurückzuführen sind [76]. Im Jahr 2008 starben 17 Millionen Menschen an Erkrankungen des Herzens und der Gefäße, was 31% der Tode weltweit ausmacht [61] und in Europa Kosten von 196 Milliarden Euro im Jahr verursacht.[1] Auch in den nächsten Jahren werden diese Erkrankungen von erheblicher gesundheitspolitischer Relevanz sein.

Die primäre Fragestellung dieser Arbeit ist, mit *welchen Methoden Ärzte Skizzen nutzen können, um den Patienten über Erkrankungen und Behandlungen von Gefäßen aufzuklären*. Die Hauptinspiration stellt die Arbeit von Zhu et al. [101] dar, welche indiziert, dass Skizzen mit der gleichzeitigen Darstellung von Flussinformationen in Bereichen wie der Biologie, dem Ingenieurwesen und der *Medizin* nützlich sein können. Während in [101] Methoden für eine weiträumige Anwendung bezogen auf Fachgebiete beschrieben werden, soll sich die vorliegende Arbeit im Speziellen mit der Blutfluss-Simulation in skizzierten Gefäßen im Kontext der Patientenaufklärung auseinandersetzen. Dadurch ist es möglich auf Besonderheiten und Anforderungen auf Arzt- und Patientenseite einzugehen. Dem Arzt wird ein Kommunikationsmittel zur Verfügung gestellt, welches ihn dabei unterstützt, dem Patienten Abläufe zu erklären. Der Patient kann Erkrankungen und Eingriffe besser verstehen und sich so mental besser auf die Behandlung vorbereiten.

Während im Fokus die *Patientenaufklärung* steht, sind weitere Anwendungsszenarien denkbar. Die *Dokumentation* der Diagnose, Therapie und Nachsorge, die der Arzt behandlungsbegleitend durchführt, kann von bildhaften Darstellungen durch

1 Die wirtschaftlichen Kosten wurden den *European Cardiovascular Disease Statistics* aus dem Jahr 2012 entnommen, siehe http://www.ehnheart.org/cvd-statistics.html (zuletzt aufgerufen: 26.02.2014).

(a) (b)

Abbildung 1: In (a) ist ein Dozent dargestellt, der eine Gefäßstruktur zeichnet. In (b) sind Kreidezeichnungen von Gefäßen dargestellt (Quelle: [71]).

Skizzen profitieren. Ein weiteres Szenario ist die *Ausbildung von Medizinern*. Medizin-Studierende lernen anatomische Strukturen häufig über Skizzen. Neben der Darstellung in Lehrbüchern werden diese häufig vom Dozenten während der Vorlesung angefertigt (siehe Abb. 1). Hierbei ist nicht nur die fertige Skizze entscheidend, welche Zusammenhänge und Funktionen vereinfacht darstellt, sondern auch der Zeichenprozess. Die Dozenten nutzen hierfür einen interaktiven, schrittweisen Ansatz, in welchem sie die einzelnen anatomischen Strukturen zeichnen, deren Funktionen erklären, weitere Strukturen zeichnen und Zusammenhänge erläutern. Hierdurch lernen die Studierenden die Funktion einzelner Strukturen sowie die Skizzen zu reproduzieren [71]. Ein weiteres Anwendungsgebiet ist die *Kollaboration*. Ärzte könnten untereinander diskutieren oder Forscher aus anderen Wissenschaftsbereichen mit einbeziehen. Durch solche Synergien ist es möglich, Gefäße und deren Strukturen besser zu verstehen oder Begründungen für das Fließverhalten von Blut zu erhalten. Weiterhin könnten die entwickelten Methoden nützlich für die *Rekonstruktion von Gefäßen* sein. Zunächst werden Daten von Medizinern akquiriert, woraufhin Techniker daraus 3D-Modelle generieren. Hierbei ist eine Kommunikation der Mediziner und Techniker hilfreich, welche von einem geeigneten Werkzeug profitieren könnten.

Aus diesen Anwendungsszenarien lassen sich zwei Ziele für die Arbeit ableiten:

1. Geeignete Methoden finden mit denen Gefäßstrukturen, relevante Erkrankungen und Behandlungsmethoden skizziert werden können. Unter geeignet wird verstanden, dass die Methoden leicht anzuwenden sind und nachvollzogen werden können.

2. Die plausible, verständliche Darstellung und echtzeitfähige Integration von Blutfluss in die Skizze.

Zur Erreichung dieser Ziele müssen folgende Aufgaben erfüllt werden:

- Eigenschaften von Gefäßen bestimmen,
- relevante Gefäßkrankheiten und Behandlungsmethoden ermitteln,
- Benutzungsschnittstelle schaffen, welche möglichst natürliches Skizzieren ermöglicht,

- Kriterien für die Eignung der Methoden bestimmen und
- echtzeitfähige Möglichkeiten zur Simulation von Blutfluss ermitteln.

Eine Einschränkung dieser Arbeit ist die Darstellung der Skizzen und des Blutflusses in 2D. Die Darstellung von Gefäßen und dem inneren Blutfluss stellt in 3D eine hohe Herausforderung an die Visualisierung dar [26]. Die zusätzliche Simulation des Blutflusses in Echtzeit würde komplexere Anforderungen darstellen. Daher wird eine zweidimensionale Darstellung, die den Vorteil bietet leichter interpretiert werden zu können, genutzt.

Die vorliegende Arbeit gliedert sich wie folgt:

- *Kapitel 2* beschreibt die Grundlagen und verwandte Literatur in den Gebieten der Medizin, der Blutfluss-Simulation und -Darstellung uns des Skizzierens. Zusätzlich werden Kriterien für die Untersuchung von Software-Ergonomie vorgestellt.
- In *Kapitel 3* der Arbeit wird ein Konzept beschrieben, welches sich damit auseinandersetzt, wie Blutfluss in Echtzeit simuliert und dargestellt werden kann. Anschließend werden Methoden für das Skizzieren von Gefäßen, deren Erkrankungen und Behandlungsmethoden erläutert.
- *Kapitel 4* beschreibt die Umsetzung des in Kapitel 3 beschriebenen Konzeptes.
- In *Kapitel 5* werden Methoden zur Evaluierung vorgestellt und auf den entwickelten Prototyp angewendet.
- Im *sechsten Kapitel* wird die Arbeit inhaltlich zusammengefasst und bewertet. Darauf folgt ein Ausblick auf weitere mögliche Forschungsfragen.

2

GRUNDLAGEN UND VERWANDTE ARBEITEN

Dieses Kapitel dient der Beschreibung aller nötigen Grundlagen, die für den Entwurf des Konzeptes (siehe Kapitel 3) nötig sind. Dabei werden verwandte Arbeiten beschrieben. Weiterhin wird die Arbeit inhaltlich abgegrenzt und weiterführende Literatur genannt. Das Kapitel besitzt folgenden Aufbau:

- Zunächst werden die nötigen *medizinischen Grundlagen* beschrieben. Hierbei wird auf Gefäße, deren Erkrankungen sowie deren Behandlungsmöglichkeiten eingegangen. Abschließend werden zwei für die Arbeit relevante Anwendungsbereiche aus dem medizinischen Umfeld beschrieben.
- Es folgt eine Beschreibung von Blut im Bezugsrahmen der Strömungslehre. Hierbei werden beginnend die mathematischen Grundlagen für eine *Blutfluss-Simulation* erläutert. Anschließend folgt eine Beschreibung verschiedener Möglichkeiten, den Blutfluss zu visualisieren.
- Abschließend wird das *Skizzieren* im Kontext der Benutzungsschnittstellen beschrieben. Hierbei wird zunächst ein historischer Überblick über skizzenbasierte Interaktion und deren weitere Entwicklung gegeben. Darauf folgt die Diskussion zwei verschiedener Benutzungsschnittstellen-Ansätze in Bezug auf deren Ergonomie. Aus dieser Diskussion wird ein Ansatz entwickelt, der in dieser Arbeit als Benutzungsschnittstellen verwendet wird.

2.1 MEDIZINISCHE GRUNDLAGEN

Die in diesem Kapitel beschriebenen medizinischen Grundlagen beziehen sich primär auf [76] (Gefäße und Gefäßerkrankungen) und [19] (Gefäßerkrankungen und Behandlung).

2.1.1 *Gefäße*

Die Gefäße des Menschen sind röhrenförmige Strukturen, die nach den Körperflüssigkeiten unterschieden werden, die sie transportieren. Dies sind zum einen Blut und zum anderen die Lymphe. Unabhängig von der Art zeigen größere Gefäße einen dreiteiligen anatomischen Aufbau. Dieser unterteilt sich in die

- *Tunica intima*, die innerste Schicht, welche eine glatte Oberfläche für optimalen Fluss besitzt, die

Abbildung 2: Die Blutgefäße eines Unterarms auf der Ausstellung *Körperwelten*. Mit der peripheren Blutversorgung nimmt die Größe der Arterien ab (Quelle: nach *blog.bookingisrael*[1]).

- *Tunica media*, die mittlere Schicht, welche aus glatten Muskelzellen besteht und die
- *Tunica adventitia*, die äußerste Schicht, welche aus lockerem Bindegewebe besteht, um das Gefäß in der Umgebung zu verankern.

Im Fokus dieser Arbeit stehen *Arterien*, also *Blutgefäße*, die das Blut vom Herzen weg führen. Im Gegensatz dazu beschreiben *Venen* Gefäße, die Blut zum Herzen hin führen. Arterien werden je nach Größe in folgende Typen unterteilt:

- *Aorta*: Die Hauptschlagader beschreibt die zentrale Arterie, die vom Herzen sauerstoffangereichertes Blut weiter in kleinere Arterien transportiert.
- *Arteriolen*: Diese kleinen Arterien gehen fließend durch Verlust der Tunica media in Kapillaren über.
- *Kapillaren*: Diese Haargefäße sind feine Gefäße, die mit dem Auge nicht mehr zu sehen sind. Durch die Verästelung mit Venen verbinden sie das arterielle und venöse Gefäßsystem.

In Abb. 2 ist der Übergang von großen zu kleinen Gefäßen dargestellt.

2.1.2 *Gefäßerkrankungen*

Die Angiologie beschäftigt sich als Teilgebiet der inneren Medizin mit Gefäßerkrankungen. Dabei werden die Erkrankungen nach den Gefäßen unterschieden, die sie betreffen (Arterien und Venen). Eine häufige Ursache für arterielle Erkrankungen ist die *Arteriosklerose*.[2] Sie beschreibt Ablagerungen von Blutfetten, Thromben, Bindegewebe und Kalk *in* den Gefäßwänden, genauer zwischen der Tunica intima und Tunica media (siehe Abb. 3). Die Ablagerungen führen zur Verringerung der Elastizität

1 http://blog.bookingisrael.com/wp-content/uploads/2012/08/circulatory-hand.jpg

2 Der Name Arteriosklerose leitet sich vom Begriff Sklerose ab, unter dem die Verhärtung von Organen oder Gewebe verstanden wird.

(a) (b)

Abbildung 3: (a) zeigt schematisch den Verlauf einer fortschreitenden Arteriosklerose mit einhergehender Stenose des Gefäßes (Quelle: nach *wikimedia*[3]). In (b) ist die Röntgenaufnahme einer mittelgradigen Stenose dargestellt (Quelle: *kardionet*[4]).

der Gefäßwände und häufig zu einer Verengung (*Stenose*) bis hin zum Verschluss des Gefäßes (siehe Abb. 3). Hierdurch entsteht einerseits die Gefahr der Gerinselbildung, andererseits die Gefahr der Unterversorgung betroffener Bereiche. Eine weitere Folge kann auch eine Schwächung der Gefäßwände sein, was zu einer Erweiterung und somit zu *Aneurysmen* führt. Hierbei besteht zweierlei Gefahr für die Betroffenen: zum einen kann die Gefäßwand im Aneurysma einreißen (*rupturieren*) und zum anderen kann das Blut im Aneurysma wie bei einer Stenose gerinnen und sich anschließend ablösen. Das abgelöste geronnene Blut kann dann ein Blutgefäß teilweise oder komplett verschließen und somit zu einer Embolie führen. Embolien lassen sich abhängig vom transportieren Material und ihrer Lokalität unterscheiden. Beispielsweise werden Schlaganfälle durch einen Embolus ausgelöst, der vom Herzen oder einer großen Arterie stammt. Neben dieser arteriellen Embolie tritt eine Lungenembolie auf, wenn der Embolus von der Vene nach Passage des Herzens eine Lungenarterie verstopft.

Eine besondere Form von Aneurysmen sind zerebrale (das Gehirn betreffende) Aneurysmen, da diese neben einem hohen Rupturierungsrisiko [64] in 40-60% der Fälle zum Tode führen [5]. Klassifiziert werden zerebrale Aneurysmen nach Eigenschaften wie ihrer Form oder Größe, woraus zwei Haupttypen entstehen: zum einen nach der am häufigsten auftretenden Form [24], *sakkuläre* Aneurysmen, welche aus einem Halsbereich bestehen, der in einen Aneurysmen-Sack übergeht. Zum anderen *fusiforme* Aneurysmen, welche keinen ausgeprägten Hals besitzen und spindelförmig ein Gefäß erweitern (siehe Abb. 4).

3 http://upload.wikimedia.org/wikipedia/commons/9/9a/Endo_dysfunction_Athero.PNG, zuletzt aufgerufen: 26.02.2014.

4 http://www.kardionet.com/Images/mittelgradige_Stenose.jpg, zuletzt aufgerufen: 26.02.2014.

5 http://www.mayfieldclinic.com/Images/PE-AneurRuptured_Figure1b.jpg, zuletzt aufgerufen: 26.02.2014.

(a) (b)

Abbildung 4: In (a) ist ein sakkuläres Aneurysma dargestellt. Zu erkennen ist sowohl der Halsbereich des Aneurysmas als auch dessen Aussackung. In (b) ist ein fusiformes Aneurysma dargestellt. Bei diesem Typ lässt sich kein Halsbereich identifizieren (Quelle: nach *Mayfield Clinic*[5]).

Venöse Gefäßerkrankungen wie Krampfadern (*Varizen*) oder Venenthrombosen werden nicht weiter behandelt. Für einen Überblick sei auf [6, 19, 76] verwiesen.

2.1.3 *Behandlung*

Die im Folgenden beschriebenen Behandlungsmethoden beziehen sich auf zwei im vorherigen Abschnitt genannte Gefäßveränderungen: Aneurysmen und Stenosen. In Abhängigkeit der anatomischen Zugänglichkeit und des Ortes des Gefäßes sowie dessen Erkrankung (*Pathologie*) sind verschiedene Behandlungsmethoden anwendbar. Im Folgenden werden zwei intravaskuläre Behandlungsmethoden (Stenting und Coiling) sowie eine extravaskuläre Behandlungsmethode (Clipping) auf Grundlage von [25] beschrieben.

Weiterführende Beschreibungen auch hinsichtlich der historischen Entwicklung von Behandlungsmethoden sind in [99] zu finden.

2.1.3.1 *Stenting*

Beim Stenting wird eine Gefäßstütze (der Stent) in ein Gefäß eingebracht, um es zumeist offen zu halten. Alternativ werden Stents als Stütze für Behandlungen von fusiformen und sakkulären Aneurysmen genutzt (siehe Abschnitt 2.1.3.2). Ein Stent ist eine röhrenförmige Struktur, die zumeist aus einem Gittergerüst aus Metall oder Kunstfasern besteht.

Die Grundlage für die Anwendung von Stents ist die *Ballondilatation*. Hierbei wird mittels eines Führungsdrahtes ein Katheter, auf dem ein entlasteter Ballon befestigt, ist in das Gefäß eingeführt und zur gewünschten Stelle (z. B. einer Stenose) bewegt. Der bis dahin entlastete Ballon wird nun aufgeblasen, drückt sich gegen die Gefäßwand und dehnt diese aus. Beim Stenting wird zusätzlich ein Stent auf dem Ballon befestigt, welcher die Ausdehnung des Ballons selbst nach dessen Entlastung und

Abbildung 5: Darstellung einer Ballondilatation mit einem Stent. Beginnend wird der Ballon mittels Katheter in der Stenose platziert. Darauf folgt das Aufblasen des Ballons und die anschließende Entfernung des Ballons sowie des Katheters. Der Stent behält die Form bei und stützt so dauerhaft die Gefäßwand (Quelle: nach *beliefnet*[6]).

der Entfernung des Katheters beibehält und so das Gefäß dauerhaft stützt. Der gesamte Prozess, der in Abb. 5 dargestellt ist, wird durch bildgebende Verfahren wie Computertomographie (CT) oder Magnetresonanztomographie (MRT) begleitet.

2.1.3.2 *Coiling*

Dieses Verfahren wird auch endovaskuläre Aneurysma-Okklusion genannt. Die Grundidee besteht darin, ein Aneurysma mit einem Draht (*Coil*) zu füllen, sodass der Blutfluss darin abnimmt. Der Draht ist auf einer Platinspirale aufgewickelt. Mittels Katheter wird der entwundene Draht zum Aneurysma bewegt. Anschließend wird der Platindraht in das Aneurysma geführt, woraufhin er sich windet und im Aneurysma verteilt. Nach ausreichender Füllung des Aneurysmas nimmt der Blutfluss darin so stark ab, dass das vorhandene Blut gerinnt und das Aneurysma okkludiert. Auch dieses Verfahren (dargestellt in Abb. 6) wird durch bildgebende Verfahren unterstützt.

Bei Aneurysmen, deren Halsbereich sehr groß ist sowie bei fusiformen Aneurysmen, kann der Coiling-Prozess dazu führen, dass sich Draht in das Gefäß hinein verteilt. Um dies zu vermeiden, kann die Behandlung durch einen Stent unterstützt werden (siehe Abb. 6).

2.1.3.3 *Clipping*

Das Clipping beschreibt eine extravaskuläre Behandlungsmethode, da die Gefäßerkrankung von außen behandelt wird. Die Grundidee beim Clipping ist es, das Aneu-

6 http://www.beliefnet.com/healthandhealing/images/si55551520_ma.jpg, zuletzt aufgerufen: 26.02.2014

7 http://www.mayfieldclinic.com/Images/PE-AneurCoiling_Figure4.jpg, zuletzt aufgerufen: 26.02.2014.

8 http://www.mayfieldclinic.com/Images/PE-AneurCoiling_Figure5.jpg, zuletzt aufgerufen: 26.02.2014.

9 http://www.mayfieldclinic.com/Images/PE-AneurCoiling_Figure6.jpg, zuletzt aufgerufen: 26.02.2014.

(a) (b) (c)

Abbildung 6: In (a), (b) und (c) ist der Ablauf des Coilings illustriert. In (a) wird der Coil in das Aneurysma hineingeschoben. (b) zeigt ein gefülltes Aneurysma sowie den angepassten Blutfluss, der nicht mehr in das Aneurysma hineinfließt. In (c) ist die zusätzliche Platzierung eines Stent dargestellt, um zu verhindern, dass der Coil aus dem Aneurysma austritt (Quelle: (a-c) *Mayfield Clinic*[7,8,9]).

rysma mit einer Klemme (*Clip*) zu verschließen, sodass kein Blut mehr hinein fließen kann. Nachdem der Zugang von außen zum Aneurysma geschaffen wurde, wird der Titan-Clip auf dem Halsbereich des Aneurysmas angebracht. Anschließend wird das Aneurysma punktiert, um zu prüfen, ob Blut durch den Clip strömt (siehe Abb. 7).

2.1.4 *Dokumentation von Erkrankungen und Patientenaufklärung*

Da die betrachteten *Anwendungsszenarien* dieser Arbeit die Dokumentation von Gefäßerkrankungen sowie die Aufklärung des Patienten beinhalten, werden beide Szenarien im Folgenden näher beschrieben.

Die *Dokumentation* von Befunden ist ein wichtiges Kommunikationsmedium zum fachrichtungsübergreifenden Austausch. So dokumentiert bspw. ein Radiologe die Ergebnisse der bildgebenden Verfahren, was wiederum dem behandelnden Arzt Informationen über die Pathologie liefert. Diese Dokumentation von Erkrankungen eines Patienten erfolgt zunehmend in digitaler Form [7]. Bestandteil sind neben der durch bspw. CT oder MRT gewonnenen Bilddaten auch Befundberichte. Diese beste-

10 http://www.mayfieldclinic.com/Images/PE-AneurClipping_Figure4b.jpg, zuletzt aufgerufen: 26.02.2014.

Abbildung 7: Das Aneurysma wurde am Hals mit einem Clip versehen, sodass kein Blut mehr hinein fließen kann (Quelle: *Mayfield Clinic*[10]).

hen meist aus wenig strukturiertem Fließtext [7]. In [78] wird beschrieben, dass eine Verschiebung von rein textuellem Inhalt hin zu bildbasiertem Inhalt dieser Reports zu Vorteilen für den Austausch, sowie der Patientenverpflegung und -Behandlung führen kann, weswegen bspw. Radiologen von Methoden profitieren könnten, die ihnen die Generierung solcher Bilder erleichtert.

Zusätzlich zur Dokumentation von Krankheitsverläufen nehmen Arzt-Patienten-Gespräche in der Medizin eine wichtige Rolle ein, weswegen sie im Fokus der vorliegenden Arbeit stehen. Dem Patienten sind hierbei verständlich aufbereitete Informationen zu seiner Krankheitssituation und zur geplanten Therapie wichtig [7]. Laut [46] fühlen sich 42% der Patienten nicht adäquat über ihre Behandlung und ihren Zustand informiert. Dabei kann sich effektive Patientenaufklärung positiv auf Aspekte wie die Medikationsmenge, die Dauer der Behandlung und den Krankenhausaufenthalt auswirken. Weiterhin sind informierte Patienten aktiver beim Umgang mit ihrer Gesundheit, sie sind unabhängiger vom Arzt und handeln Verantwortungsbewusster [46].

Die *Pflichten* des Behandelnden in § 630e des Bürgerlichen Gesetzbuches (BGB) festgehalten. So beschreibt § 630e Abs. 1 BGB:

> *Der Behandelnde ist verpflichtet, den Patienten über sämtliche für die Einwilligung wesentlichen Umstände aufzuklären. Dazu gehören insbesondere Art, Umfang, Durchführung, zu erwartende Folgen und Risiken der Maßnahme sowie ihre Notwendigkeit, Dringlichkeit, Eignung und Erfolgsaussichten im Hinblick auf die Diagnose oder die Therapie [...].*

Zur Aufklärung erhalten die Patienten häufig Fragebögen als Vorabinformationen. Diese können Darstellungen der Pathologie und der Behandlung enthalten. Neben dieser meist generalisierten Aufklärung findet auch eine individualisiertere Aufklärung statt, die durch § 630e Abs. 2 Nr. 1 BGB näher beschrieben wird:

> *Die Aufklärung muss mündlich durch den Behandelnden [...] erfolgen [...]; ergänzend kann auch auf Unterlagen Bezug genommen werden, die der Patient in Textform erhält.*

Durch die mündliche Aufklärung soll gewährleistet werden, dass der Patient sämtliche für die Einwilligung zur Behandlung wesentlichen Umstände zur Kenntnis genommen und verstanden hat. Der zweite Teil des Gesetzesauszugs bezieht sich auf etwaige Fragebögen, die der Patient im Vorfeld erhalten hat.

Weitere Szenarien wie das *Training* und die *Planung von Operationen* sind nicht Gegenstand der Arbeit. Für das Training ist eine präzise Darstellung der Gefäße und Pathologien üblich, um dem Lernenden ein realitätsnahes Szenario zu bieten. Dies steht im Gegensatz zu der in dieser Arbeit geplanten Abstraktion und Vereinfachung der Gefäßstrukturen. Für die Operationsplanung ergibt sich eine teilweise Überschneidung mit den Zielen dieser Arbeit: Chirurgen fertigen zur Vorbereitung der Operationen zwar Skizzen an [7], allerdings würden sich hierdurch andere Anforderungen für Skizzierungsmethoden ergeben, weshalb auch dieses Szenario nicht weiter betrachtet wird.

2.2 BLUTFLUSS-SIMULATION UND -DARSTELLUNG

2.2.1 *Simulation von Fluiden*

Unter Fluiden versteht man sowohl Flüssigkeiten als auch Gase. Das Wisschenschaftsgebiet, welches sich mit der Simulation des Verhaltens von ihnen beschäftigt, wird Computational Fluid Dynamics (CFD) genannt. Sie werden eingeteilt in newtonsche und nicht-newtonsche Fluide. Newtonsche Fluide (wie Wasser und Luft) sind dadurch gekennzeichnet, dass ihre *Schergeschwindigkeit* (räumliche Veränderung der Flussgeschwindigkeit) proportional zu ihrer *Scherspannung* (ein Maß für die Kraft pro Flächeneinheit, die bei der Scherung wirkt) ist. Der Zusammenhang zwischen diesen beiden Größen beschreibt Viskosität, also die Zähflüssigkeit eines Fluids. Die newtonschen Fluide haben aufgrund der Proportionalität dieser beiden Größen eine konstante Viskosität, unabhängig von den Scherkräften. Nicht-newtonsche Fluide (wie Treibsand) verhalten sich anders, da die Linearität zwischen Schergeschwindigkeit und Scherspannung dort nicht gegeben ist. Anders ausgedrückt können diese Fluide bei hohen Scherkräften dünn- oder dickflüssiger werden. Diese Eigenschaft ist unter anderem ein Grund für das Verhängnis eines im Treibsand Untergehenden. Befindet sich jemand im Treibsand und rührt sich nicht, sind die Scherkräfte sowie Viskosität gering, was dazu führt, dass er langsam einsinkt. Versucht sich der Einsinkende nun aus dem Treibsand zu befreien, erhöhen sich die Scherkräfte und damit auch die Viskosität des Sandes, weshalb ihn dieser fest umklammert [23]. Auch bei Blut, welches Gegenstand dieser Arbeit ist, handelt es sich um ein nicht-newtonsches Fluid. Die Viskosität von Blut nimmt im Gegensatz zu Treibsand bei hohen Scherkräften ab. Es ist somit in der Lage, auch Kapillaren mit Blut zu versorgen.[11] Eine weitere Besonderheit von Blut ist die Art, wie es durch den Körper transportiert wird. Durch die rhythmischen Kontraktionen des Herzens wird es pulsierend durch die Blutgefäße gepumpt.

Die Simulation von dynamischer Viskosität ist aufwendiger und komplizierter als die Simulation von newtonsche Fluiden. Um die gewünschte Simulation zu vereinfachen, wird eine konstante Viskosität für das zu simulierende Blut angenommen und es somit als newtonsche Flüssigkeit betrachtet. Verfahren, die sich mit der numerischen Beschreibung von nicht-newtonschen Fluiden beschäftigen, sind u. a. in [15] beschrieben.

Die folgenden weiteren Vereinfachungen werden für die Simulation angenommen: Das Fluid ist *homogen* und *inkompressibel*. Hierdurch ist es möglich, die später genauer beschriebenen Navier-Stokes-Gleichungen zur Darstellung des Zustands des Fluids anzuwenden. Die Eigenschaft der Homogenität bedeutet, dass die Dichte des Fluids konstant im gesamten Raum ist. Ist ein Fluid inkompressibel, so bedeutet dies, dass sein Volumen in jeder Unterregion über die Zeit konstant ist. Die Schulphysik lehrte uns die Unterscheidung zwischen Gasen und Flüssigkeiten *halbwahr*: Gase können ihr Volumen ändern, Flüssigkeiten hingegen nicht. Da Blut eine Flüssigkeit ist, wäre demnach die genannte Vereinfachung für die Simulation von Blut nicht nötig. Bei der Darstellung der Schulphysik handelt es sich allerdings um ei-

11 Die geringeren Scherkräfte ergeben sich durch die Verformung der roten Blutkörperchen. Diese verändern ihre Gestalt in den Kapillaren von scheibchenförmigen zu länglicheren Gebilden, wodurch die Viskosität abnimmt.

ne *Lüge für Kinder*[12]: Eine vereinfachte bzw. falsche Darstellung, um dennoch ein Grundverständnis zu vermitteln, um später darauf aufzubauen. Tatsächlich sind alle Fluide, wenn auch verschieden stark, kompressibel. Wäre dies nicht so, wären bspw. Wale nicht in der Lage im Wasser zu kommunizieren, da Schall zur Übertragung ein kompressibles Medium braucht. Die Kompressibilität von Wasser ist allerdings sehr gering und die Simulation von dieser, mit welcher sich der Zweig der *Compressible flows* (siehe [21]) beschäftigt, ist kompliziert und rechenaufwendig [9]. Trotz dieser Vereinfachung können echtzeitfähige plausible und anschauliche Ergebnisse bei der Simulation erzielt werden. Dies ist auch, im Gegensatz zur physikalisch korrekten Darstellung von Blut, das primäre Ziel dieser Arbeit. Die Kombination aus Homogenität *und* Inkompressibilität bedeutet, dass die Dichte bezogen auf die Zeit und den Raum konstant ist.

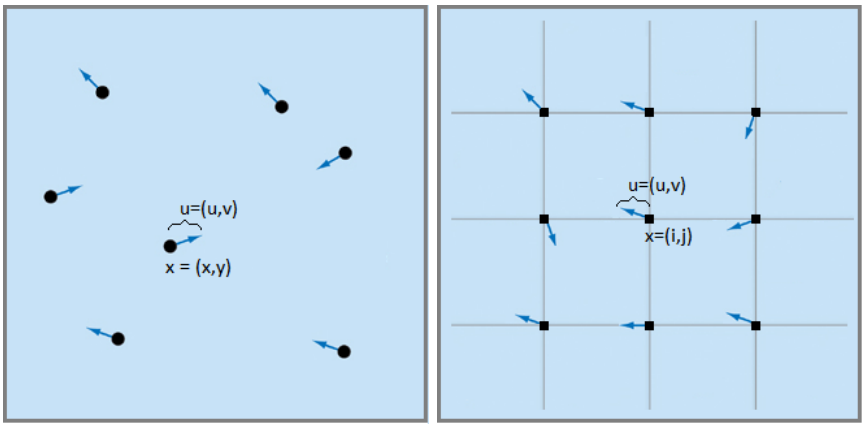
Bei der geplanten Simulation von Blutfluss in Gefäßen werden weiterhin nur zwei Typen von Stoffen betrachtet: das Fluid selbst und ein Feststoff (engl. solid). Alternativ wäre es möglich, mehrere verschiedene Fluide mit unterschiedlichen Eigenschaften (z. B. Wasser und Luft) zu simulieren. Die Grenze, an der diese beiden Fluide aufeinandertreffen, wird als Freiflächengrenze (engl. freesurface) bezeichnet. Thematisiert wurden Freiflächengrenzen bereits 1965 von Harlow und Welch [31]. Ein echtzeitfähiges Verfahren, welches die Level-Set-Methode zur Umsetzung nutzt, ist in [69] beschrieben.

Um eine Flüssigkeit zu simulieren, ist es nötig, deren Zustand darzustellen. Dabei sind verschiedene Betrachtungsweisen möglich. Zwei davon sind die

- *lagrangesche* Betrachtungsweise und die
- *eulersche* Betrachtungsweise.

In Abbildung 8 sind die beiden verschiedenen Darstellungen veranschaulicht. Bei der Umsetzung der Lagrange-Variante muss ein Partikelsystem definiert werden. Diese Form der Fluidsimulation wird bspw. bei der geglätteten Teilchen-Hydrodynamik (engl. Smoothed Particle Hydrodynamics (SPH)) eingesetzt, um bspw. das Verhalten von Tsunamis zu berechnen [85]. Einen Überblick über diese Methode ist in [55] zu finden. Beispiele für die Verwendung von SPH im Zusammenhang mit Blutflussvisualisierungen sind in [62] und [77] zu finden. Beide Arbeiten beschäftigen sich mit der Partikel-basierten Darstellung von Blutfluss innerhalb von Gefäßen, um einen Chirurgen in virtuellen Operationssystemen zu unterstützen (siehe 9). Die hier genutzte Euler-Darstellung wird verwendet, da die gitterbasierte Betrachtung weit verbreitet ist und Methoden dafür ausführlich in der Literatur beschrieben sind. Diese werden zusammen mit den Navier-Stokes-Gleichungen für inkompressible, homogene Flüssigkeiten im nächsten Abschnitt beschrieben. Die Gleichungen und deren Lösung basieren dabei grundlegend auf der Arbeit *Stable fluids* von Stam [87] und lehnen sich an [9] und [32] an. Da die gewünschte Darstellung zweidimensional ist, werden die Formeln und Lösungen entweder allgemein oder für den zweidimensionalen Fall beschrieben.

12 *Lie-to-children* - Der Begriff wird von Ian Stewart und Jack Cohen u. a. in [16] erklärt und verwendet.



(a) Lagrange-Verfahren (b) Euler-Verfahren

Abbildung 8: Gegenüberstellung der lagrangeschen und eulerschen Darstellung eines Fluids: (a) Der Zustand des Fluids wird durch Partikel dargestellt. Jeder Partikel besitzt eine Position $x = (x, y)$ und einen Geschwindigkeitsvektor $u = (u, v)$. Die Partikel bewegen sich im Laufe der Zeit entlang ihrer Geschwindigkeit und verändern so den Zustand des Fluids. (b) Der Zustand des Fluids wird durch ein fixiertes Gitter dargestellt. Die relevanten Größen (wie Geschwindigkeit) und deren Veränderung in der Zeit werden an diesen fixierten Punkten untersucht.

(a) (b)

Abbildung 9: Darstellung von Blutfluss in Gefäßen: In (a) beschreiben die Partikel den Blutfluss. Dabei werden sich schnell bewegende Partikel gelb und sich langsam bewegende Partikel rot gefärbt (Quelle: [62]). In (b) wird die Gefäßwand mit einem Masse-Feder-System modelliert und kann so auf den Blutfluss reagieren. Zu sehen ist ein rupturierendes Aneurysma, aus dem Blut austritt (Quelle: [77]).

2.2.1.1 *Navier-Stokes-Gleichungen*

MATHEMATISCHE GRUNDLAGEN Die Navier-Stokes-Gleichungen beinhalten Differentialoperatoren, also Abbildungen, die einer Funktion eine andere Funktion zuordnen. Das Ergebnis der hier verwendeten Operatoren enthält dabei die Ableitung nach einer oder mehreren Variablen. Im Folgenden werden die in den Gleichungen verwendeten Operatoren beschrieben.

Gradient Der Gradient ist ein Differentialoperator, der auf einem Skalarfeld angewendet wird und einen Vektor als Ergebnis liefert. Das Skalarfeld kann bspw. durch eine Funktion $f : \mathbb{R}^n \to \mathbb{R}$ beschrieben werden. Der Ergebnisvektor besteht dann aus den partiellen Ableitungen dieser Funktion. Im zweidimensionalen Fall ist der Gradient, der durch den Nabla-Operator ∇ beschrieben wird, folgendermaßen definiert:

$$\nabla f(x, y) = (\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}).$$

Der Ergebnisvektor zeigt dabei in Richtung des steilsten Anstiegs; der Betrag des Vektors ist ein Maß für den Anstieg.

Divergenz Der Divergenzoperator $\nabla\cdot$ kann auf Vektorfelder angewendet werden und erzeugt dabei als Ergebnis ein Skalarfeld. Für den zweidimensionalen Fall ist er folgendermaßen definiert:

$$\nabla \cdot \vec{u} = \nabla \cdot (u, v) = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}.$$

Hierbei wird er auf ein Vektofeld angewendet und liefert ein Skalarfeld. Die Schreibweise $\nabla\cdot$ erklärt sich dadurch, dass symbolisch zwischen dem Gradienten und dem Vektorfeld das Skalarprodukt gebildet wird, also:

$$\nabla \cdot \vec{u} = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}) \cdot (u, v) = \frac{\partial}{\partial x} u + \frac{\partial}{\partial y} v.$$

Würde man den Divergenzoperator auf verschiedene Punkte eines Strömungsfeldes eines fließenden Baches anwenden, so würde der entstehende Skalar beschreiben, ob es sich beim ausgewählten Punkt um eine Quelle handelt (Skalar größer Null), eine Senke (Skalar kleiner Null) oder einen Punkt, an dem ebenso viel Wasser abwie zufließt (Skalar gleich Null). Wenn der Operator auf einem Vektorfeld für jeden Punkt die Divergenz Null liefert, so bezeichnet man das Vektorfeld als divergenzfrei.

Laplace Der Laplace-Operator Δ wird gebildet, indem die Divergenz $\nabla\cdot$ auf den Gradienten ∇ angewendet wird. Daraus ergibt sich die alternative Schreibweise ∇^2, die im Folgenden verwendet wird. Wird der Operator auf ein Skalarfeld angewendet, resultiert daraus ebenfalls ein Skalarfeld. Für die zweidimensionale Funktion $f : \mathbb{R}^2 \to \mathbb{R}$ ist er folgendermaßen definiert:

$$\nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}.$$

Verwendung findet der Laplace-Operator z. B. bei der Rekonstruktion von Polygonnetzen. Setzt man das Ergebnis der Gleichung Null ($\nabla^2 f = 0$) entsteht die namensgebende Laplace-Gleichung. Ist das Ergebnis der Gleichung ungleich Null ($\nabla^2 f \neq 0$) so wird diese Poisson-Gleichung genannt.

Finite-Differenzen-Methode Differentialgleichungen und somit auch die vorgestellten Operatoren lassen sich mithilfe des Computers nicht ohne Weiteres lösen, da diese mit stetigen Funktionen definiert werden. Um das Problem dem Computer zugänglich zu machen, ist eine Diskretisierung nötig. Diese wird erreicht, indem das Gebiet, für das eine Funktion gelten soll, in endlich viele Gitterpunkte unterteilt wird; eine zweidimensionale Funktion wird so in ein Rechteckgitter unterteilt. An diesen Gitterpunkten können nun die Ableitungen approximiert werden. Die Finite-Differenzen-Methode (FDM) beschreibt numerische Verfahren einer solchen Approximation. Sie leitet sich aus links- und rechtsseitigen Differenzenquotienten der Taylorreihe ab. Für diese Arbeit relevant ist eine Kombination aus beiden: Der sogenannte *zentrale Differenzenquotient*. Genaueres zur Herleitung der FDM aus der Taylorformel und der Taylorreihe kann in [9] nachgelesen werden. Die Bildung des zentralen Differenzenquotienten wird durch folgende Formel beschrieben:

$$\frac{\partial q}{\partial x}(x_i) = \frac{q_{i+1} - q_{i-1}}{2\Delta x}.$$

q ist dabei eine eindimensionale Funktion, x beschreibt den Parameter nach dem differenziert wird, x_i ist die Stelle an der die Ableitung bestimmt werden soll, $q_{i+1} \coloneqq q(x_{i+1})$, $q_{i-1} \coloneqq q(x_{i-1})$ und Δx ist der Abstand zweier benachbarter Gitterpunkte. Aus dieser Vorschrift ergibt sich die Berechnung des Gradienten, der Divergenz sowie des Laplace-Operators, die im Folgenden näher beschrieben werden.

FDM für den Gradienten

$$\nabla p = \left(\frac{p_{i+1,j} - p_{i-1,j}}{2\Delta x}, \frac{p_{i,j+1} - p_{i,j-1}}{2\Delta y}\right).$$

$p = p(i,j)$ ist ein zweidimensionales Skalarfeld und Δx sowie Δy sind die Abstände zweier benachbarter Gitterpunkte in x- und y-Richtung.

FDM für die Divergenz

$$\nabla \cdot u = \frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x} + \frac{v_{i,j+1} - v_{i,j-1}}{2\Delta y}.$$

wobei $u = (u, v)$ ein zweidimensionales Vektorfeld ist.

FDM für den Laplace-Operator

$$\nabla^2 p = \frac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{(\Delta x)^2} + \frac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{(\Delta y)^2}.$$

Bei einem kartesischem Gitter, welches für die Simulation genutzt wird, sind Δx und Δy gleich. Dies vereinfacht die FDM-Vorschrift für den Laplace-Operator zu:

$$\nabla^2 p = \frac{p_{i+1,j} + p_{i-1,j} - 4p_{i,j} + p_{i,j+1} + p_{i,j-1}}{(\Delta x)^2}. \tag{1}$$

In dieser Vorschrift ist eine Analogie zur Bildverarbeitung erkennbar. Zur Kantendetektion wird der sogenannte *five-point stencil* genutzt, welcher durch die Matrix

$$\begin{pmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{pmatrix}$$

dargestellt wird und den Gewichtungen von (1) entspricht.

Bei der Anwendung der FDM muss die Größe des Gitters begrenzt werden. An diesen Grenzen gelten andere Berechnungsvorschriften, die durch die *Randwertbedingungen* bestimmt werden. Diese werden im Kapitel 2.2.1.1 im Zusammenhang mit Fluiden genauer beschrieben werden.

GLEICHUNGEN Die Navier-Stokes-Gleichungen beschreiben die Grundgleichungen der Stömungsmechanik für newtonsche Fluide. Sie bestehen aus nichtlinearen, partiellen Differentialgleichungen zweiter Ordnung und beruhen auf den physikalischen Erhaltungssätzen. Die beiden Gleichungen für inkompressible, homogene Flüssigkeiten können folgendermaßen beschrieben werden:

Gleichung 1

$$\frac{\partial u}{\partial t} = -(u \cdot \nabla)u - \frac{1}{\rho}\nabla p + \nu\nabla^2 u + F. \tag{2}$$

Hierbei ist u ein zweidimensionales Geschwindigkeitsvektorfeld, t ist die Zeit, ρ ist eine Konstante für die Dichte, p ist ein Skalarfeld des Drucks, ν ist ein Parameter für die Viskosität und F stellt externe Kräfte als Vektorfeld dar. Diese Gleichung ist die *Impulsgleichung* (engl. *momentum equation*), da sie die einwirkenden Kräfte und die Reaktion des Fluids darauf beschreibt. Sie kapselt die Advektion, die Diffusion, den Druck sowie die von außen einwirkenden Kräfte. Diese Begriffe werden mit ihren zugehörigen Termen in den folgenden Abschnitten einzeln genauer betrachtet. Die zweite Gleichung stellt eine Bedingung dar:

Gleichung 2

$$\nabla \cdot u = 0, \tag{3}$$

wobei u das zweidimensionale Vektorfeld aus der ersten Gleichung ist. Diese *Kontinuitätsgleichung* (engl. *continuity equation*) sichert Inkompressibilität, indem vorausgesetzt wird, dass das Vektorfeld u divergenzfrei sein muss.

ADVEKTION 28800 Badespielsachen wurden am 10. Januar 1992 im pazifischen Ozean von einem Lieferungsschiff transportiert. Während des Transports kam es zu einem schweren Sturm, woraufhin Container vom Schiff fielen und die Badespielsachen im Ozean verteilten. Sie wurden von der Strömung im pazifischen Ozean an die Küsten von Hawaii, Alaska, Canada und weiterer Orte getrieben[13] (siehe Abb. 10).

Der Vorgang, der die u. a. enthaltenen Spielzeug-Enten durch die Strömung an verschiedene Orte treibt, wird *Advektion* genannt.

In der Fluidsimulation wird die Strömung durch das Geschwindigkeitsfeld u repräsentiert. Durch u können beliebige Größen advektiert werden (z. B. im Fluid platzierte Tinte) oder die Geschwindigkeit selbst. Von der Selbstadvektion hängt auch ab,

13 Donovan Hohn begab sich 15 Jahre nach dem Verlust der Fracht auf die Suche nach den Badespielsachen und hielt seine Ergebnisse in *Moby Duck* [36] fest.

14 http://upload.wikimedia.org/wikipedia/commons/thumb/1/1f/Friendly_Floatees.png/800px-Friendly_Floatees.png, zuletzt aufgerufen: 26.02.2014.

Abbildung 10: Das Frachtschiff startete in Hongkong und verlor die Badespielsachen am Endpunkt des schwarzen Pfeils. Über die Jahre tauchten diese an verschiedenen Orten auf (Quelle: *Wikipedia*[14]).

ob ein stetiges Fluid oder ein unstetiges Fluid simuliert wird. Bei stetigen Fluiden ändern sich Parameter wie Geschwindigkeit, Dichte oder Druck während der Simulation nicht. Da das Geschwindigkeitsfeld dynamisch sein soll, handelt es sich um die Simulation eines unstetigen Fluids. Folgender Term der Navier-Stokes-Gleichung beschreibt die Selbstadvektion der Geschwindigkeit:

$$-(u \cdot \nabla)u = -(u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y}) \begin{pmatrix} u \\ v \end{pmatrix}.$$

Bei der Advektion der Badespielsachen handelt es sich um die Lagrange-Betrachtung, da die Ente einen Partikel repräsentiert. Diese steht im Gegensatz zur hier verwendeten Euler-Betrachtung, erleichtert aber die Beschreibung vom sog. Semi-Lagrange-Schema für Advektion. Die Semi-Langrange-Advektion nimmt an, dass sich ein Partikel an jedem Gitterpunkt befindet. Nun wird die Frage gestellt, an welcher Position sich dieser Partikel an einem vorherigem Zeitpunkt befand. Diese wird ermittelt, indem der Geschwindigkeitsvektor am Gitterpunkt bestimmt und von der Gitterkoordinate abgezogen wird. Von der so bestimmten neuen Position wird der Geschwindigkeitsvektor ermittelt und für den Partikel bzw. der Gitterposition als neuer Geschwindigkeitsvektor genommen (siehe Abb. 11). Der Name Semi-Lagrange-Schema ergibt sich aus der Kombination aus eulerscher und lagrangescher Betrachtungsweise.

Die beschriebene Methode enthält zwei Komponenten, die im Folgenden näher diskutiert werden:

- Die Verfolgung eines Partikels zu einem früheren Zeitpunkt und
- die Bestimmung des Geschwindigkeitsvektors an der ermittelten Position.

Die Verfolgung wird im Kontext der Vektorfelder *Integration* genannt. Das beschriebene Verfahren, die Euler-Integration, ist eine simple und schnell zu berechnende

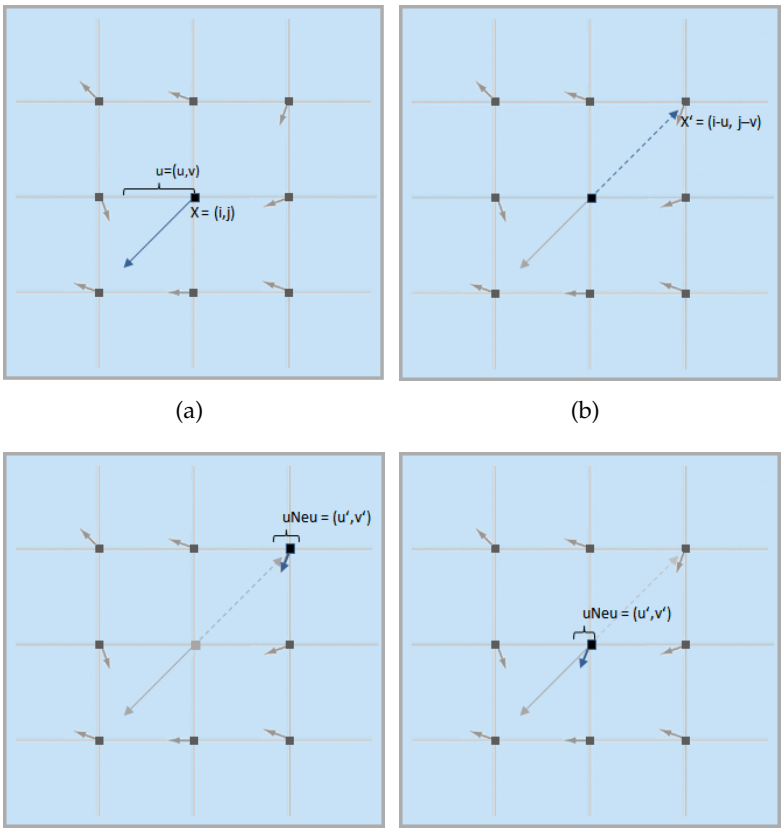


Abbildung 11: Illustration des Semi-Lagrange-Verfahrens mittels Euler-Integration für einen Gitterpunkt.

Methode. Bei Vektorfeldern, die eine starke Krümmung aufweisen, neigen die berechneten Positionen allerdings zu einem *Abdriften*. Würde man die Euler-Integration wiederholt auf einem kreisförmigen Strömungsfeld anwenden, so würde sich die ermittelte Position immer weiter vom Zentrum entfernen, wobei ein Beibehalten des Abstands zum Zentrum gewünscht ist. Eine Reihe von Verfahren, die dieses Verhalten erzielen, sind unter dem Begriff Runge-Kutta-Integration zusammengefasst. Hierbei wird die Bewegung des Partikels iterativ in mehreren Stufen bestimmt [10].

Die durch das Integrationsverfahren bestimmte Position kann an drei Stellen liegen:

- genau auf einem Gitterpunkt
- außerhalb des Gitters
- zwischen mehreren Gitterpunkten

Liegt die Position genau auf einem Gitterpunkt, so kann der Geschwindigkeitsvektor von diesem einfach übernommen werden. Befindet sich die ermittelte Position außerhalb des Gitters, so wird die neue Position abhängig von den Randwertbedingungen bestimmt (siehe Abschnitt 2.2.1.1). Tritt der dritte Fall ein, so muss die neue Geschwindigkeit aus den Nachbarn gebildet werden. Zur Bestimmung sind verschiedene Verfahren möglich, wie die Verwendung des *nächsten Nachbarn*, *bilineare Interpolation* oder eine *Interpolation höherer Ordnung* (z. B. Hermiteinterpolation, siehe [20]). Der Vorteil bei der Verwendung des nächsten Nachbarn oder der Verwendung von linearer Interpolation liegt darin, dass der ermittelte Wert zwischen den Werten der Nachbarn liegt. Hierdurch wird das Verfahren stabil, was zur Namensgebung von Stams *Stable fluids* führt. Problematisch ist, dass dieses Interpolationsverfahren einen Mittelwert der vorangegangenen Geschwindigkeiten bildet und nach häufiger Anwendung zu einer Glättung der Geschwindigkeitsvektoren führt. Dieser Effekt wird *numerical dissipation* bzw. *numerical diffusion* genannt und kann durch Interpolationsverfahren höherer Ordnung vermieden werden. Bei der Verwendung dieser besteht das Risiko, dass die neu ermittelten Werte über oder unter den vorhandenen liegen, was die Simulation unstabil macht.

DRUCK Wenn eine Kraft auf einen Punkt im Fluid wirkt, ist diese nicht sofort im gesamten Volumen sichtbar. Stattdessen wirkt diese zuerst am Ursprungspunkt und breitet sich allmählich aus. Wie schnell sich diese Kraft ausbreitet, wird mit dem Druck beschrieben, welcher von Regionen mit hohem Druck hin zu Regionen mit niedrigerem Druck wirkt. Von Interesse ist die Beschleunigung dieser Kraft, welche mit der ersten Ableitung beschrieben wird. Für ein Vektorfeld kann diese somit durch Anwendung des Gradienten erhalten werden. Da die Wirkung von hohem zu niedrigerem Druck dargestellt wird, muss das Vorzeichen angepasst werden. Aus diesen Beschreibungen folgt der den Druck beschreibende Term aus der Navier-Stokes-Gleichung $-\nabla p$.

EXTERNE KRÄFTE Durch externe Kräfte werden die von außen auf das System einwirkenden Kräfte modelliert. Diese können global wirken, wie die Gewichtskraft, die auf das gesamte Vektorfeld nach unten wirkt oder eine lokale Kraft, die das Vektorfeld an bestimmten Stellen manipuliert. Diese Kräfte werden durch das Vektorfeld F in den Navier-Stokes-Gleichungen beschrieben

DIFFUSION Die Diffusion beschreibt einen physikalischen Prozess, mit der sich zwei oder mehrere Stoffe selbst ohne das Einwirken von externen Kräften vermischen. Das bedeutet, dass sich zwei verschieden gefärbte, ruhende Flüssigkeiten im gleichen Behälter nach und nach ineinander ausbreiten, bis sie vollständig vermischt sind. Hierbei übt die Viskosität der Flüssigkeiten einen starken Einfluss auf die Vermischungsgeschwindigkeit aus. Zähflüssige Fluide vermischen sich langsamer als dünnflüssige.

In den Navier-Stokes-Gleichungen wird die Diffusion durch den Term $\nu\nabla^2 u$ beschrieben. Stellt man diesen Term nach $\nabla^2 u = \frac{1}{\nu} u$m, so entspricht er einer Poisson-Gleichung, für dessen Berechnung das Lösen eines linearen Gleichungssystems erforderlich ist. Dies ist mithilfe der Verwendung eines *Solvers* möglich, welcher durch ein numerisches Verfahren eine Lösung schätzt, die über die Anzahl von Iterationsschritten genauer berechnet werden kann. Da die Echtzeitfähigkeit und Interaktivität der Simulation im Vordergrund steht, muss beim Lösen des linearen Gleichungssystems ein Kompromiss zwischen Berechnungsgeschwindigkeit und Genauigkeit des Ergebnisses gefunden werden. Eine leichte Diffusion wird unabhängig vom Diffusionsterm simuliert, da sie durch die in Abschnitt 2.2.1.1 beschrieben *numerical diffusion* entsteht. Die Glättung der Geschwindigkeit wirkt wie die Kraft, die der Resistenz eines Partikels gegen die Bewegung ihrer Nachbarn entspricht [9]. Beispiele für Verfahren zum Lösen des linearen Gleichungssystems sind das Jacobi-Verfahren (siehe [32]) oder das Gauß-Seidel-Verfahren (siehe [48]), welche sich in Bezug auf Berechnungsaufwand und Konvergenzgeschwindigkeit des Ergebnisses unterscheiden. Diese und weitere Verfahren werden in [48] diskutiert.

BERECHNUNG Werden die beschriebenen Berechnungen der Advektion, Diffusion und externen Kräfte auf das Geschwindigkeitsfeld angewendet, so entsteht ein neues Vektorfeld, welches im Folgenden w genannt wird. Dieses Vektorfeld ist aufgrund der Operationen nicht divergenzfrei, was aber als Bedingung durch die zweite Navier-Stokes-Gleichung gegeben ist. Um aus w ein divergenzfreies Vektorfeld zu machen, ist das Helmholtz-Theorem hilfreich. Dieses besagt, dass ein Vektorfeld w folgendermaßen in ein Vektorfeld und ein Skalarfeld zerlegt werden kann:

$$w = u + \nabla p, \tag{4}$$

wobei u ein *divergenzfreies* Vektorfeld ist (ein Beweis ist in [14] zu finden). Das Theorem ermöglicht somit die Korrektur der Divergenz von w, indem der Gradient eines Skalarfeldes von w abgezogen wird. Dieser Gradient entspricht dem Skalarfeld des Drucks p. Das divergenzfreie Vektorfeld lässt sich also durch Umstellen der Gleichung 4 nach u erhalten.

Nun fehlt noch die Berechnungsvorschrift für das Vektorfeld des Drucks p. Auch hierbei ist das Helmholtz-Theorem hilfreich. Wenn der Divergenzoperator auf beide Seiten von Gleichung 4 angewendet wird, erhält man:

$$\nabla \cdot w = \nabla \cdot (u + \nabla p) = \nabla \cdot u + \nabla^2 p.$$

Da die zweite Navier-Stokes-Gleichung besagt, dass $\nabla \cdot u = 0$ ist, vereinfacht sich Gleichung und man erhält die Berechnungsvorschrift für p:

$$\nabla^2 p = \nabla \cdot w.$$

Dies ist wiederum eine Poisson-Gleichung, die durch das Lösen eines linearen Gleichungssystems gelöst werden kann. Mit dem divergenten Vektorfeld w kann also p gefunden werden, wodurch das neue, divergenzfreie Vektorfeld, u bestimmt werden kann (siehe Gleichung 4).

RANDWERTBEDINGUNGEN Bei den FDM sind Randwertbedingungen nötig. Dies sind Vorschriften dafür, wie sich die Grenzen des definierten Gebiets bei der Approximation der Differentialoperatoren verhalten. Bei der verwendeten Simulationsmethode sind die Randwertbedingungen für zwei Komponenten nötig: zum einen für das Geschwindigkeits-Vektorfeld und zum anderen für das Druck-Skalarfeld. Weiterhin wird das Fluid in einem rechteckigen Gitter simuliert. Am Rand dieses Gitters soll keine Flüssigkeit austreten können. Für die Geschwindigkeit ergibt sich somit, dass sie am Rand auf Null abfällt. Dieses Festlegen des Wertes wird als *Dirichlet*-Randwertbedingung bezeichnet.

Für den Druck ist die Berechnung komplexer. Hier ist eine *Neumann*-Randwertbedingung nötig, die nicht für eine Funktion einen Wert festlegt, sondern für deren Ableitung. Hierbei gilt folgende Berechnungsvorschrift: $\frac{\partial p}{\partial n} = 0$; in der Richtung der Normale muss die Ableitung von p Null sein.

Im Advektionsabschnitt 2.2.1.1 wurde für das Semi-Lagrange-Schema der Fall genannt, dass die bestimmte Position außerhalb des Gitters liegt. In diesem Fall wird angenommen, dass der Partikel im vorherigem Zeitschritt auf der Grenze lag. Somit wird die Geschwindigkeit folgend der Dirichlet-Randwertbedingung auf Null gesetzt.

2.2.2 *Skalar- und Vektorfelddarstellung*

Der vorherige Abschnitt beschreibt Gleichungen, die auf zweidimensionalen Skalar- und Vektorfeldern operieren. Hierdurch ergeben sich an dem diskretisierten Gitter an jedem Punkt mehrere Parameter, die dargestellt werden können. Eine wichtige darzustellende Größe ist die im Advektionsabschnitt 2.2.1.1 genannte Tinte, die im Strömungsfeld platziert wird und dem Nutzer so Informationen über die Strömung liefert; hierbei handelt es sich um ein Skalarfeld. Außerdem soll das Strömungsfeld selbst dargestellt werden, welches aus einem zweidimensionalen Vektorfeld besteht. Die dafür nötigen Visualisierungsmethoden sind der wissenschaftlichen Visualisierung zuzuordnen, da physikalische Daten dargestellt werden. Dennoch kann ein Ziel der Informationsvisualisierung entlehnt werden: das Gewinnen von Erkenntnissen und Einsichten [13, 75]. Der folgende Abschnitt beschäftigt sich mit Methoden zur Darstellung von Skalar- und Vektorfeldern.

2.2.2.1 *Skalarfeldvisualisierung*

Die skalaren Werte der Tinte, welche in der sonst einfarbigen Flüssigkeit advektiert werden, können auf verschiedene Arten dargestellt werden. Da sich die Werte des skalaren Feldes zu jedem Zeitschritt ändern können, ist eine Technik erforderlich, die diese Veränderung berücksichtigt. Eine der häufigsten Techniken sind Kontur- bzw. Isolinien [47]. In [59] werden Möglichkeiten zur Isolinienberechnung sowie Techniken zur Darstellung von Isolinien auf zeitveränderlichen Skalarfeldern vorgestellt.

Andere Techniken zur Visualisierung ergeben sich, wenn das Problem vom Standpunkt der Informationsvisualisierung betrachtet wird. Bei den Informationen in einem Skalarfeldes handelt es sich um trivariate Daten: Zwei Attribute werden durch die x- sowie y-Koordinate repräsentiert und ein Attribut ist der Skalar, dessen Ausprägung sich im Zeitverlauf ändert. Die x- und y-Komponente kann nun auf den Achsen eines Koordinatensystems dargestellt werden und der Skalar durch bspw. Farbe, Form oder Größe. Der Skalar kann so z. B. als Radius eines Kreises in einem Blasendiagramm wie der *Gapminder Trendalyzer*[15] visualisiert werden. Eine weitere weit verbreitete und häufig angewendete Darstellung ist die Kodierung des skalaren Wertes durch Farbe. Um dem Nutzer eine einfache Einschätzung der Größe zu ermöglichen, muss eine geeignete Farbskala angewendet werden. Nach einer ersten Überlegung könnte die im Visualisierungsbereich am häufigsten verwendete Farbskala genutzt werden: die Regenbogenskala [8]. Eine Veränderung des skalaren Wertes wird bei ihr durch eine Veränderung des Farbtons dargestellt. Dabei ist der Vergleich von Farbtönen keine geeignete Methode, um Distanzen in skalaren Werten abzulesen, da in verschiedenen Farbtönen keine perzeptuelle inhärente Ordnung existiert[16] [8, 54]. Besser geeignet ist daher eine z. B. einfarbige Skala, die von schwarz (kleiner Wert) zu einer Farbe (großer Wert) verläuft. Ein Vorteil der einfarbigen Skala ist, dass der Nutzer leicht Unterschiede in der Größe erkennen kann und so Werte vergleichen kann. Eine konkretere Auseinandersetzung mit der Nutzung einer Farbskala ist in Kapitel 3.1.3 zu finden.

2.2.2.2 *Strömungsfeldvisualisierung*

Die Sichtbarmachung von Strömung ist bereits lange Gegenstand der Forschung. Ein sehr frühes Beispiel sind die Arbeiten von Leonardo da Vinci (1452-1519). Er interessierte sich stark für Wasser und dessen Strömungsverhalten. Seine Ergebnisse sind neben zahlreichen Notizen, in denen er u. a. die fließende Bewegung des Wassers mit der Bewegung von Haaren vergleicht [56], auch Skizzen, in denen er Einfluss von Objekten auf fließendes Wasser sowie Verwirbelungen darstellt (siehe Abb. 12).

Eine spätere Motivation zur Darstellung von Strömungen besteht bspw. in der Wettervorhersage, beim Entwurf von Fluggeräten oder in der Automobilindustrie. Eine Technik zum Sichtbarmachen sind Woll- und Textilfäden (engl. tufts). Diese werden auf den zu untersuchenden Strömungskörper angebracht und anschließend unter dem Einfluss eines starken Windes beobachtet. Hierdurch können lokale Strömungsphänomene sichtbar gemacht werden (siehe Abb. 13). Diese Technik wurde von Shen und Pang [83] in das Virtuelle übertragen; dort werden virtuelle Fäden auf das 3D-Modell angebracht und durch die Strömung verändert. Ein Vorteil dieser Methode ist neben der Bekanntheit des Verfahrens aus seinem realem Vorbild, dass die virtuellen Fäden die Strömung dort nicht beeinflussen. Weitere analoge Verfahren sind in [66] zu finden.

15 http://www.gapminder.org, Gapminder Trendalyzer stellt Trends animiert in ihrem Zeitverlauf durch ein Blasendiagramm dar.

16 Die Farbtöne der Regenbogenskala können zwar z. B. nach ihren Wellenlängen sortiert werden, diese Sortierung kann im Sinne von größer oder kleiner vom Menschen allerdings nicht wahrgenommen werden. Bei der Aufgabe, einzelne Farbtöne auf Platten zu ordnen, variierten die Anordnungen von Testperson zu Testperson; einige sortierten die Farben sogar alphabetisch [8].

17 http://upload.wikimedia.org/wikipedia/commons/9/98/Studies_of_Water_passing_Obstacles_and_falling.jpg, zuletzt aufgerufen: 26.02.2014.

(a) (b)

Abbildung 12: In (a) ist eine Strömung dargestellt, die auf ein Hinderniss trifft und sich dort teilt. In (b) ist einfließendes Wasser und die daraus entstehenden Wirbel illustriert (Quelle: nach Wikipedia[17]).

Abbildung 13: Auf beiden Autos sind Wollfäden angebracht. Während in (a) durch das Ausfransen der Fäden starke Verwirbelungen erkennbar sind, liegen die Wollfäden in (b) glatt am Auto an (Quelle: [66]).

Für die digitale Darstellung von Vektorfeldern ist eine Vielzahl von Techniken bekannt. Nur eine Auswahl davon ist für diese Arbeit relevant, da das Strömungsfeld besondere Anforderungen stellt. Durch die beschriebene Selbstadvektion der Geschwindigkeit ergibt sich die Simulation eines unstetigen Fluids. Weiterhin kann sich in jedem Zeitschritt der Zustand des Fluids ändern. Somit ist ein zweidimensionales Visualisierungsverfahren für unstetige, zeitabhängige Fluide gesucht.

Im folgenden Abschnitt werden Klassen von Visualisierungsverfahen vorgestellt und jeweils Vertreter daraus genannt und beschrieben. Die Klassifikation folgt dabei [30] und [96] und unterteilt sich in

- direkte,
- stellenweise und
- dichte Strömungsvisualisierungen.

Auf eine weitere Klasse sei der Vollständigkeit halber verwiesen: Die merkmalsbasierte Strömungsvisualiserung. Diese wird nicht näher erläutert, da diese in dieser Arbeit keine Anwendung findet. Eine Übersicht über merkmalsbasierte Verfahren ist in [72] zu finden. Eine umfangreiche Übersicht über Verfahren zur Vektorfeldvisualisieurung inklusive Erläuterungen, Darstellungen und Quelltext stellt Zhanping Liu zur Verfügung.[18]

DIREKTE STRÖMUNGSVISUALISIERUNG Direkte Strömungsvisualisierung ist die einfachste Form zur Sichtbarmachung von Strömungen. Ein bekanntes Beispiel für eine direkte Strömungsvisualisierungstechnik sind Pfeilplots, welche Glyphen zur Darstellung der Strömung verwenden (siehe Abb. 14(a)). Die Pfeile werden an den Punkten des Gitters gezeichnet. Diese entsprechen dabei nicht zwangsläufig dem zugehörigen Punkt im Gitter, sondern können auch lokale Stellvertreter eines Bereiches sein. Mit der Richtung der Pfeile wird die Flussrichtung beschrieben. Wird zusätzlich die Kraft der Strömung durch die Länge des Pfeils dargestellt, handelt es sich um den sog. Hedgehog-Ansatz (siehe [73]). Diese Methode ermöglicht das direkte Visualisieren von zeitabhängigen Vektordaten. Zusätzliche Eigenschaften wie Beschleunigung, Krümmung oder lokale Rotation können durch Glyphen repräsentiert werden [18]. Vorteilhaft an diesen Methoden ist ihre einfache Umsetzung sowie die Möglichkeit zur schnellen Berechnung [60].

Wird eine hohe Anzahl von Pfeilen dargestellt, führt dies zu einer hohen kognitiven Last, was problematisch ist, da so das Gesamtbild der Strömung nicht wahrgenommen werden kann. Durch Clustern von Datenpunkten kann die Wahrnehmung der zugrundeliegenden Daten verbessert werden. Hierbei kann allerdings eine Problematik von Pfeilplots verstärkt werden: wichtige Elemente und Bereiche des Feldes werden evtl. nicht dargestellt.

STELLENWEISE DARSTELLUNG VON PARTIKELVERFOLGUNG Bei der Gruppe dieser Techniken werden charakteristische Linien durch die Verfolgung einzelner Partikel im Strömungsfeld gewonnen (siehe Abb. 14(b)). Hierbei gibt es verschiedene Typen von Linien, die bei unstetigen Vektorfeldern unterschieden werden:

18 http://www.zhanpingliu.org/research/flowvis/FlowVis.htm, zuletzt aufgerufen: 26.02.2014.

- *Pathlines* beschreiben den Pfad, den ein verfolgter Partikel während des Zeitverlaufs nimmt.
- *Streamlines* werden *an einem* bestimmten Zeitpunkt erzeugt. Das Vektorfeld wird also „eingefroren“ und anschließend wird ein Partikel ausgesetzt und verfolgt.
- *Streaklines* beschreiben Partikel, die kontinuierlich über eine Zeitspanne an *einer Stelle* losgelassen werden. Die verbundene Anordnung dieser Partikel zu einem Zeitpunkt beschreibt die Streakline.
- *Timelines* werden erzeugt, indem an einem bestimmten Zeitpunkt Partikel an *verschiedenen Orten* erzeugt werden. Der Verlauf dieser Partikel bildet die Timeline.

Da bei dieser Klasse Geometrie in Form von Linien erzeugt wird, wird diese Klasse von Techniken auch geometrische Strömungsvisualisierung genannt [60].

Eine grundlegende Herausforderung besteht darin, geeignete Saatpunkte zu finden, an denen Partikel losgelassen werden können [96]. Hiervon hängt ab, ob relevante und kritische Eigenschaften des Vektorfeldes abgebildet werden. Zur Bestimmung der Saatpunkte sind generell zwei Verfahren möglich:

- Sie können durch das *System* oder
- den *Nutzer* festgelegt werden.

Wenn das System die Saatpunkte bestimmt, besteht eine Möglichkeit darin, diese gleichmäßig über das Strömungsfeld zu verteilen (siehe [94, 42]). Ein Beispiel für eine Arbeit für nutzerbestimmte Saatpunkte ist in [80] zu finden. Dort werden vom Nutzer Linien in ein Strömungsfeld skizziert. An Punkten auf dieser skizzierten Linie werden Saatpunkte für Streamlines gesetzt und die Linien verfolgt. Anschließend wird überprüft, welcher Streamline die gezeichnete Linie am ähnlichsten ist und mit einer Animation wird die skizzierte Linie an die Streamline angenähert. Auf diese Weise besitzt der Nutzer eine starke Kontrolle darüber, wie das zugrundeliegende Strömungsfeld durch Streamlines dargestellt wird.

Eine weitere Fragestellung ergibt sich für die Anwendung eines geeigneten Integrationsverfahrens, welches bestimmt, wie Partikel im Strömungsfeld verfolgt werden. Hierbei existieren mehrere Möglichkeiten, wie in Abschnitt 2.2.1.1 bereits erwähnt wird. Ein Vergleich von Integrationsverfahren bzgl. Berechnungsaufwand und Genauigkeit ist in [92] zu finden; eine aktuelle Zusammenfassung stellt [60] dar.

Der Vorteil von stellenweiser Darstellung von Partikelverfolgung ist, dass wichtige Elemente des Strömungsfeldes dargestellt werden können, ohne den gesamten Bereich auszufüllen. Weiterhin ist durch die Variation der verschiedenen Linientypen eine vielfältige Darstellung der Eigenschaften des Vektorfeldes möglich. Für zeitveränderliche Daten besteht die Schwierigkeit darin, von Zeitschritt zu Zeitschritt eine framekohärente Darstellung zu erhalten. In [43] wird ein Verfahren beschrieben, welches diese Problematik für Streamlines adressiert. In jeweils zwei Zeitschritten werden zunächst korrespondierende Streamlines identifiziert. Diese werden bei der Simulation von einem Zeitschritt in den jeweils nächsten interpoliert (durch sog. *Morphing*), so dass der Eindruck entsteht, es handle sich um eine Streamline, die sich durch das Vektorfeld bewegt.

(a) direkte Darstellung (b) vereinzelte Darstellung (c) dichte Darstellung

Abbildung 14: In (a) ist ein Pfeilplot dargestellt; durch die Farbe wird die Stärke eines Vektors kodiert (Quelle: *Zhanpingliu*[19]). (b) zeigt gleichmäßig verteilte Streamlines (Quelle: [42]). In (c) ist die LIC-Methode dargestellt (Quelle: [50]).

DICHTE DARSTELLUNG VON PARTIKELVERFOLGUNG Diese Technik wird auch texturbasierte Strömungsvisualisierung genannt, da flächenfüllende Texturen als Grundlage verwendet werden (siehe Abb. 14(c)). Die Methoden der stellenweise und dichten Darstellung sind eng miteinander verwandt, da beide auf Partikeladvektion beruhen. Abhängig vom Dichtegrad der dargestellten Linien der Partikel gehen diese beiden Techniken fließend ineinander über.

Eine bekannte und etablierte Technik einer dichten Darstellung von Partikelverfolgung ist Line Integral Convolution (LIC) [12]. Hierbei wird eine aus weißem Rauschen bestehende Textur generiert, die anschließend mit dem Strömungsfeld lokal geglättet (bzw. verschmiert) wird. Bei dem so erzeugtem Bild wird allerdings weder die Stärke, noch die Richtung der Vektoren angezeigt. Um dies zu erreichen existieren angepasste Methoden, in denen bspw. die Stärke durch eine Farbe kodiert wird [86] oder die Richtung durch eine Animation [22]. Für die Darstellung von unstetigen Vektorfeldern sollte eine Methode angewendet werden, die frame-kohärente Bilder erzeugt. Neben [22], bei dem dies durch Animation erreicht wird, gibt es die Technik der *dynamischen Line Integral Convolution* (kurz DLIC), die in [88] beschrieben wird. Hier wird die Änderung von dem Vektorfeld eines Zeitschritts zum Vektorfeld des nächsten Zeitschritts als Bewegungsvektorfeld gespeichert. Mithilfe dieses Bewegungsvektorfeldes und üblicher LIC werden Übergänge erzeugt, die dem Nutzer flüssig erscheinen.

Ein Überblick über weitere texturbasierte Verfahren ist in [50] und [79] zu finden. In [50] werden Vor- und Nachteile ausgewählter Methoden diskutiert. [79] stellt eine Übersicht dar und unterscheidet dabei zwischen stetigen und unstetigen Techniken sowie zwischen zwei- und dreidimensionalen Verfahren.

Ein Vorteil von der dichten Darstellung von Partikelverfolgung gegenüber der stellenweisen Darstellung ist, dass die Problematik des Findens der Saatpunkte nicht gegeben ist. Weiterhin können diese Techniken von der Berechnung durch die Graphics Processing Unit (GPU) profitieren, da diese parallel viele Operationen gleichzeitig ausführen kann. Zusätzlich beinhalten die resultierenden Darstellungen viele Details und stellen so charakteristische Eigenschaften des Strömungsfeldes wie z. B. Wirbel dar.

Abbildung 15: Interaktion mit dem Lightpen auf einem Röhrenmonitor (Quelle: *Wikipedia*[20]).

2.3 SKIZZIEREN

Menschen nutzen in vielen Domänen Skizzen als einfache und schnelle Möglichkeit, Ideen zu kommunizieren [44]. So werden Skizzen beispielsweise dazu genutzt, erste Prototypen von Anwendungen zu entwerfen und mit Nutzern zu testen. Hierdurch können Ideen erprobt werden und dabei der hohe Aufwand für die Implementierung eines funktionierenden Programms vermieden werden [11]. Durch Skizzen können komplexe Sachverhalte ausgedrückt werden, ohne dabei einen starken Fokus auf Präzision und Genauigkeit zu legen. Diese Stärke kann jedoch gleichwohl eine Schwäche sein, die in Verbindung mit dem Computer auftritt. Soll dieser das Skizzierte verarbeiten und interpretieren, so wird dies durch Ungenauigkeiten erschwert [44]. Dennoch bietet die Kombination von Skizzen und dem Computer ein großes Potential: Sie ermöglicht es eine interaktive Oberfläche zu gestalten, die sich im einfachsten Fall wie ein Blatt Papier verhält, gleichzeitig aber intelligenter ist [82], da es das Interagieren mit dem Gezeichneten ermöglicht.

Die Grundlage für das Skizzieren am Computer stellt der sog. Lichtgriffel (engl. *Lightpen*) dar. Bis zu seiner Erfindung wurde der Computer durch das Drücken von Knöpfen und die Eingabe von Befehlen bedient. Der Lightpen ermöglichte eine neue Art der Interaktion direkt am Röhrenbildschirmen, wobei die Position des Lightpens auf dem Bildschirm bestimmt wurde (siehe Abb. 15). Er wurde im Jahre 1948-1949 entwickelt [38] und die Interaktionstechnik stark durch das Militär gefördert.[21] Eines der ersten Programme, die den Lightpen zu komplexerer Interaktion nutzte, ist

19 http://www.zhanpingliu.org/research/flowvis/Arrows/Arrow_Reg_FixSize_CM.GIF, zuletzt aufgerufen: 26.02.2014.

20 http://commons.wikimedia.org/wiki/File:HypertextEditingSystemConsoleBrownUniv1969.jpg, zuletzt aufgerufen: 26.02.2014.

21 In den 1950ern wurde das SAGE (Semi-Automatic Ground Environment), ein Luftverteidigungssystem, entwickelt. Bedient wurde es durch Air-Force-Offiziere (und keine Computerexperten), weshalb die Bedienung möglichst einfach sein sollte. Die Aufgabe der Soldaten war es feindliche Bomber zu identifizieren, die auf einem Röhrenbildschirm durch Lichtpunkte dargestellt wurden. Hielten sie ein

(a) (b)

Abbildung 16: In (a) ist erkennbar, dass für ein gezeichnetes Liniensegment nur der Start- und Endpunkt relevant ist, nicht aber der gesamte Pfad. In (b) ist der Kreiszeichenmodus dargestellt. Anhand des Zeichnungspfades wird durch das System ein Kreis berechnet, der dargestellt wird. Das Umschalten der Zeichenmodi geschieht in Sketchpad durch Drücken von Knöpfen (Quelle: nach [89]).

Sketchpad. Es wurde 1963 von Sutherland entwickelt und in seiner Dissertation beschrieben [89]. Sketchpad ist ein Zeichentool, welches Liniensegemente, Kreise und Symbole erstellen, editieren und löschen kann (Beispiele sind in Abb. 16 dargestellt). Diese Arbeit stellt einen fundamentalen Beitrag im Bereich der Mensch-Computer-Interaktion (engl. Human Computer Interaction (HCI)) dar und beschreibt eine der ersten grafischen Benutzungsoberflächen (engl. Graphical User Interface (GUI)) [90]. Sketchpad legte damit eine Grundlage sowohl für die heutige Interaktion mit dem Computer als auch für Eingabegeräte wie die Maus, das Grafiktablett und aktuelle touch-basierte Eingaben mit Smartphones oder Tablets [44]. Einhergehend mit dieser Entwicklung wurden verschiedene Arten von Benutzungsschnittstellen entwickelt sowie Paradigmen für die Nutzerinteraktion. Im Folgenden werden zwei dieser Konzepte, die für diese Arbeit relevant sind, näher beschrieben und gegenübergestellt.

2.3.1 *WIMP-Paradigma*

Das WIMP-Paradigma beschreibt ein Grundkonzept von GUIs [95]. Laut [41] stellt es einen Schritt in der Evolution der Benutzungsschnittstellen dar, nimmt dort somit eine wichtige Position ein und wird daher im Folgenden näher erläutert. Die Grundlage des WIMP-Paradigmas stellen die vier namensgebende Komponenten dar:

- *Fenster* (engl. *w*indows) stellen visuelle Bereiche dar, welche z. B. Programme, Warnhinweise oder Bearbeitungswerkzeuge enthalten. Sie sind variabel in ihrer Position und Größe und können sich gegenseitig überdecken.
- *Symbole* (engl. *i*cons) sind kleine Piktogramme, welche eine Datei, ein Programm oder ein Kommando darstellen können. Geeignete Icons können stell-

Flugobjekt für verdächtig, wurde es mit dem Lightpen berührt und damit wurde dem Computer der Befehl gegeben, diesen Lichtpunkt zu verfolgen [57].

vertretend für eine textuelle Beschreibung eingesetzt werden, wodurch Platz eingespart wird sowie zur Entstehung von mentalen Modellen beitragen [75].

- *Menüs* (engl. *m*enu) stellen dem Nutzer meist durch ausklappbare Listen mehrere Auswahlmöglichkeiten bereit. Dabei wählt der Nutzer Funktionen aus, die textuell oder ikonisch beschrieben sind.

- *Zeiger* (engl. *p*ointer) ist ein grafischer Repräsentant, der die Bewegung des Zeigegerätes abbildet. Er kann genutzt werden, um Elemente des GUIs zu selektieren oder zu verschieben. Hierdurch ergibt sich implizit das Grundkonzept der *direkten Manipulation*. Der Nutzer kann mit den grafischen Repräsentanten interagieren und erhält dabei *unmittelbar* eine Rückkopplung über die ausgeführte Aktion [75, 84].

Die Vorteile von WIMP-GUIs sind, dass die Bedienung von ihnen schnell zu erlernen, die Benutzung einfach und über Anwendungen hinweg ein Wissenstransfer[22] möglich ist. Nachteilig ist, dass die Nutzer viel Zeit damit verbringen, die Benutzungsschnittstelle kennenzulernen und zu verstehen, anstatt das Programm zu nutzen. Für Experten ergibt sich durch die Verwendung von vielen Symbolen für Funktionen häufig Frustration, da sie Tastenkürzel präferieren [95].

Weitere Nachteile betreffen die *Zeiger*-Komponente. Diese ist nötig, da das verwendete Eingabegerät (wie die Maus oder ein Grafiktablett) ein *indirektes Zeigegerät*[23] darstellt. Die indirekt-gekoppelte Steuerung des Zeigers ist dabei unnatürlich und führt zu einer höheren kognitiven Belastung. Im Gegensatz dazu stehen *direkte Eingabegeräte* wie stift- oder touch-basierte Eingaben direkt auf der Ausgabefläche. Diese werden vom Benutzer als natürlicher angesehen, da sie der Nutzung aus der realen Welt (z. B. das Schreiben mit Stift und Papier) ähneln. Daher eignet sich für das Skizzieren ein direktes Zeigegerät. Bei direkten Eingabegeräten ist der Nachteil zu erwähnen, dass der Nutzer durch bspw. den Stift oder die Hand Bereiche des Bildschirms verdeckt. Eine Konsequenz aus den Schwächen des WIMP-Paradigmas ist deren Weiterentwicklung. GUIs mit dieser Zielstellung nennen sich *Post-WIMP-GUIs* und werden im Folgenden näher erläutert.

2.3.2 *Post-WIMP*

Laut Dam [95] muss es bei Post-WIMP-GUIs mindestens eine Interaktionstechnik geben, die nicht von Elementen des WIMP-Paradigmas abhängig ist. So geschieht die Eingabe bei Post-WIMP-GUIs statt mit Menüs und Symbolen bspw. durch Gesten und Spracherkennung. Ein Beispiel für ein Post-WIMP-GUI sind Fahrsimulatoren, welche zur Eingabe Lenkrad, Gaspedal und Schaltknüppel nutzen. Während die Lernkurve solcher Benutzungsschnittstellen anfangs steiler als bei WIMP-GUIs sein kann, ist es bei geübteren Nutzern möglich, die kognitive Last zu reduzieren [95]. In [41] werden die vielfältigen Interaktionstechniken von Post-WIMP-GUIs unter einem Konzept zusammengefasst: der Reality-Based Interaction (RBI). Diese Bezeichnung rührt aus

22 Bedienkonzepte, die in einer Anwendung erlernt wurden, können auch bei der Nutzung eines anderen Programms genutzt werden.

23 Indirekte Zeigegeräte sind dadurch gekennzeichnet, dass sich die Eingabe- und Ausgabefläche unterscheidet.

der Überlegung, das die neuartigen Interaktionsstile von vorhandenem Wissen der Nutzer profitieren. Dieses Wissen stammt dabei aus der täglichen, nicht digitalen Welt [41]. So kann die Vertrautheit der Nutzer auf Papier zu zeichnen am Computer genutzt werden. Zwei Beispiele dafür sind in [1] und [63] beschrieben. In [1] wurde ein User Interface (UI) zum Skizzieren von geometrischen Objekten für Designer entworfen, welches dem Nutzer die Bedienung sowohl durch Elemente des WIMP-Paradigmas als auch des Post-WIMP-Paradigmas (in Form von Freihandzeichnung und Objekterkennung durch den Computer) ermöglicht. Eine Befragung der Nutzer führte zu dem Ergebnis, dass sie den skizzenbasierten Ansatz präferierten. In [63] wird eine Benutzungsschnittstelle zum Skizzieren von parametrischen Kurven beschrieben. Dabei wird das WIMP-Paradigma hinsichtlich der nötigen Anzahl der Interaktionen zum Zeichnen von Linien und Kurven mit einem Sketch-based Interface (SBI) verglichen. Es wurde gezeigt, dass selbst bei Fehlern in der Skizzenerkennung mit dem SBI ein Effizienzgewinn erzielt werden kann. Diese Arbeiten bieten Hinweise darauf, dass SBIs Vorteile gegenüber klassischen UIs besitzen, weshalb diese folgend näher erläutert werden.

2.3.3 *Sketch-based Interface*

Die Grundlage für ein SBI bildet das Eingabegerät, welches dem Nutzer ermöglichen muss, *Freihand*-Eingaben zu tätigen. Da diese Definition z. B. Maus und Grafiktabletts einschließen würde, wird sie erweitert. Das Eingabegerät muss zusätzlich ein *direktes Zeigegerät* sein (siehe vorheriger Abschnitt) [17].

Das Zeigegerät liefert dem zugrundeliegenden System Positionsinformationen in einem zweidimensionalem Koordinatensystem. Durch Unterschiede in den Abtastraten von verschiedenen Eingabegeräten und durch den irregulären Abstand dieser Positionen durch eine variierende Zeichengeschwindigkeit des Nutzers entstehen Herausforderungen bei der Verarbeitung. In [81] werden die Unregelmäßigkeiten bei der Eingabe als *Rauschen* beschrieben und auf zwei Quellen zurückgeführt: auf

- ungenaue Eingaben des Nutzers und auf
- Quantisierung.

Die erste Quelle resultiert daraus, dass der Nutzer eventuell nicht sauber zeichnet oder seine Hand (leicht) zittert. Die zweite Quelle betrifft die Digitalisierung einer stetigen Bewegung. Selbst bei hohen Abtastraten können bei sehr schnellen Bewegungen nur wenig Positionsinformation auf einer bestimmten Fläche erhalten werden [81]. Wünschenswert im Hinblick auf die spätere Verarbeitung wären hingegen möglichst *wenige* Positionsinformationen, die aber weiterhin den Skizzierungsverlauf repräsentieren sowie Positionen, die *gleichmäßig* verteilt sind. Um diese zu erhalten, ist eine Nachverarbeitung der Eingabe nötig, die durch folgende Schritte erreicht wird:

1. Neuabtastung (engl. resampling) und Glätten,
2. Verschönerung (engl. beautification),
3. Erkennung (engl. basic recognition).

Resampling dient dazu, die abgetasteten Punkte gleichmäßig zu verteilen. Durch anschließendes Glätten können die erhaltenen Punkte beim Verbinden „weicher“ wirken. Für das Resampling sowie das Glätten werden konkrete Methoden im Konzeptteil (siehe 3.2.1) beschrieben. Eine komplexere Form des Resamplings wird durch das sog. *Fitting* erreicht. Hierbei wird nicht versucht, die Abtastpunkte möglichst gleichmäßig zu verteilen, sondern die Anzahl weiter zu reduzieren. Beim Fitting wird versucht, das Gezeichnete durch parametrische Kurven wie *Bézierkurven* [70] oder *Splines* [4] zu beschreiben.

Die *Verschönerung* der Eingabe beschreibt eine größere Veränderung des Gezeichneten. In [39] wird ein Ansatz für interaktive Verschönerung vorgestellt. Der Nutzer skizziert dabei Linienzüge, die das Programm verarbeitet und nach *geometrischen Bedingungen* mit bisherigen Linienzügen vergleicht. Die geometrischen Bedingungen betreffen Rechtwinkligkeit, Kongruenz (z. B. Parallelität) oder Symmetrie. Nach dem Zeichnen einer Linie schlägt das Programm dem Nutzer verschiedene Interpretationen der Linie vor, von welchen der Nutzer eine auswählt.

Bei der *Erkennung* wird das Gezeichnete mit einer internen Repräsentation von Symbolen abgeglichen. Falls die Ähnlichkeit einen gewissen Schwellwert erreicht, kann das Gezeichnete vom System als dieses Symbol interpretiert werden [44]. Hierdurch ist es z. B. möglich geometrische Formen wie Kreise zu erkennen und die Skizze durch einen Kreis zu ersetzen. Weiterhin ermöglicht die Verwendung eines *Recognizers* die Verwendung von *Gesten*. Für eine einfache und effiziente Methode zur Umsetzung von Erkennungsfunktionen sei auf den *1¢ Recognizer*[24] [35] verwiesen. Durch Gesten können die beim WIMP-Paradigma auftretenden Störungen des Interaktionsflusses durch *Moduswechsel* vermieden werden: Anstatt dem Nutzer für jeden Modus ein Symbol anzubieten, welches er auswählen kann, können Moduswechsel durch verschiedene Gesten herbeigeführt werden. Hierdurch erhöht sich allerdings der Lernaufwand der Anwendung.

Eine wichtige Anwendungsdomäne von SBIs sind Modellierungsaufgaben. Diese Benutzungsschnittstellen werden unter dem Namen Sketch-based Interface for modeling (SBIM) zusammengefasst. Das Ziel von SBIMs ist es Freihandskizzen beim Modellierungsprozess von 3D-Objekten sowohl für grobe Modellierungsaufgaben als auch für detaillierte Konstruktion zu nutzen. In [67] wird eine Taxonomie von SBIMs nach drei Kriterien vorgestellt:

- 3D-Modelle erstellen
- Details hinzufügen und
- Modelle bearbeiten.

Für diese drei Kriterien werden zahlreiche Systeme vorgestellt und verglichen. Eine Arbeit im medizinischen Kontext stellt [71] dar. Dort wird ein SBIM für das Skizzieren und Modellieren von sich verzweigenden Gefäßen beschrieben. Die Basis bilden hierbei Konventionen, die durch die Untersuchung von anatomischen Zeichnungen gefunden wurden. Nach dem Skizzieren werden die Oberflächen der 3D-Modelle mithilfe von impliziten Oberflächen erzeugt.

24 Der *1¢ Recognizer* ist ein Nachfolger des bekannten *$1 recognizers* [98]. Der kleinere Betrag soll die einfachere Implementierung und die mindestens so gute Erkennrate des *$1 recognizers* aufzeigen.

Eine andere Fragestellung von SBIM betrifft die Darstellung der 3D-Objekte. Anstatt diese mit üblichen Techniken wie z. B. einem Phong-Shading darzustellen, können nicht-photorealistisches Rendering (NPR)-Techniken genutzt werden, um den Skizzencharakter zu erhalten [95]. Hier eignen sich illustrative Techniken, wie z.B *Hatching* [74] oder kontur- und feature-basierte Streamlines [51]. Ein Vergleich solcher Techniken ist in der Arbeit von Lawonn et al. [52] zu finden.

2.3.4 *Usability und User Experience*

Bei verschiedenen Bedienungskonzepten, wie das WIMP-Paradigma bzw. SBIs, stellt sich die Frage, inwieweit ein derart umgesetztes System ergonomisch für einen Nutzer zu verwenden ist. Das Teilgebiet der HCI, welches sich mit dieser Frage auseinandersetzt, ist die *Software-Ergonomie* bzw. *Usability*. Hierbei wird der Benutzer in den Kontext der Nutzung des Systems gestellt. Usability zielt darauf ab, Systeme zu entwickeln, die aufgabenangemessen, nützlich und gut bedienbar sind [75]. Folgende Ziele sind durch die normierte[25] Software-Ergonomie geregelt:

- *Selbstbeschreibungsfähigkeit* kennzeichnet, wie sehr die Benutzungsoberfläche kommuniziert, welche Aktionen vom Nutzer ausgeführt und *wie* diese ausgeführt werden können.
- *Aufgabenangemessenheit* stellt die Frage, ob der Nutzer seine Aufgaben effektiv und effizient erledigen kann.
- *Steuerbarkeit* beschreibt die Steuerung von Abläufen durch den Nutzer. Hierbei ist relevant, ob ein Nutzer seine momentane Aufgabe unterbrechen und später fortsetzen kann oder inwieweit Aktionen rückgängig gemacht bzw. wiederholt werden können.
- *Erwartungskonformität* kennzeichnet, wie sehr die Anwendung den Erwartungen des Nutzers oder allgemeinen Konventionen entspricht. Weiterhin wird damit beschrieben, wie konsistent dessen Bedienung ist.
- *Fehlertoleranz* bezeichnet die Eigenschaft eines Systems auf unvorhergesehene Eingaben oder Fehler in der Hard- und Software zu reagieren.
- *Individualisierbarkeit* kennzeichnet, wie sehr das System an die Bedürfnisse und Kenntnisse des Nutzers anpassbar ist.
- *Lernförderlichkeit* beschreibt, ob die Anwendung den Nutzer beim Erlernen von Funktionen unterstützt und anleitet.

Ein weiteres Kriterium, welches nicht in der Norm enthalten ist, aber für die Akzeptanz und die Motivation der Benutzer entscheidend ist, ist die *User Experience* [75]. Dieses bezieht ästhetische Aspekte in das interaktive System mit ein. Die genannten Ziele müssen begleitend zur Entwicklung eines Prototyps beachtet werden, da die Ergebnisse einer späteren Betrachtung schwierig oder aufwendig umzusetzen sind [37].

25 Die Norm EN ISO 9241 beschreibt Richtlinien für die HCI. Im Teil 110 sind *Grundsätze der Dialoggestaltung* wie Aufgabenangemessenheit, Selbstbeschreibungsfähigkeit usw. erläutert.

Nachdem diese Taxonomie für die Benutzbarkeit interaktiver Systeme vorgestellt wurde, sind Methoden nötig, die Usability zu untersuchen. Hierbei werden nach [37] zwei Möglichkeiten unterschieden: *Inspektions*-Methoden (welche ohne Endnutzer angewendet werden) und *Test*-Methoden (welche mit Endnutzern angewendet werden). Ein Beispiel für eine Inspektionsmethode ist der sog. *cognitive Walktrough* [97]. Hierbei wird ein Usability-Experte zurate gezogen, welcher sich in einen Endanwender hineinversetzt und konkrete Handlungsabläufe analysiert. Die Usabilty-Test-Methoden hingegen, welche mit Endnutzern durchgeführt werden, sind zur Überprüfung der Software-Ergonomie unverzichtbar, da sie direkte Informationen darüber liefern, wie Nutzer ein System verwenden und welche Probleme sie damit haben. Diese Methoden, welche aus bspw. der Think-Aloud-Methode, Feldstudien und Fragebögen bestehen, werden in dieser Arbeit verwendet und genauer im Evaluierungskapitel (siehe 5) beschrieben. Ein Überblick über weitere Methoden aus beiden Bereichen ist in [37] zu finden.

2.3.5 *Betrachtung vom WIMP-Paradigma und von SBIs unter Berücksichtigung der Usability*

Sowohl Elemente des WIMP-Paradigmas als auch SBIs besitzen Vor- und Nachteile, die im Zusammenhang mit den Kriterien der Usability folgend genauer untersucht werden.

SELBSTBESCHREIBUNGSFÄHIGKEIT Während ein auf dem WIMP-Paradigma basierendes System die möglichen Funktionen bildhaft durch Symbole und textuell durch Menüs beschreibt, bietet ein SBI häufig nur eine reine Skizzierungsoberfläche, welche weniger selbstbeschreibend ist.

AUFGABENANGEMESSENHEIT Abhängig von den Aufgaben der Anwendung eignet sich das WIMP-Paradigma oder ein SBI eher für deren Ausführung. Beispielsweise können Moduswechsel bei einer geringen Anzahl von Modi *effektiv* durch Betätigen von Symbolen ausgeführt werden. Bei einer hohen Anzahl von Modi und somit Symbolen nimmt die *Effizienz* allerdings ab, da der Nutzer das entsprechende Symbol suchen bzw. lange Wege mit bspw. der Maus zwischen ihnen zurücklegen muss. In SBIs werden Moduswechsel häufig mit Gesten durchgeführt. Bei einer hohen Anzahl von Modi steigt zwar die kognitive Last des Nutzers, da er sich viele Gesten merken muss. Diese können allerdings im gesamten Zeichenbereich ausgeführt werden, was eine *effektive* und *effiziente* Ausführung ermöglicht.

STEUERBARKEIT In WIMP-basierten UIs werden Dialoge und Aktionen durch Klicken auf Symbole oder direkte Manipulation von Objekten gesteuert. In SBIs werden hierfür hingegen häufig Gesten genutzt. Die genaue Steuerung der Abläufe ist hier stark von der jeweiligen Umsetzung abhängig, weshalb beide Ansätze gleichermaßen geeignet sind.

ERWARTUNGSKONFORMITÄT Dieses Kriterium hängt stark mit den Erfahrungen des Benutzers zusammen. Durch den bereits genannten Wissenstransfer sind viele Nutzer mit dem WIMP-Paradigma vertraut und können ihre bereits gelernte Interak-

(a) (b)

Abbildung 17: In (a) ist ein *marking Menü* dargestellt. Dieses wird sichtbar, nachdem der Nutzer auf ein Objekt tippt. Anschließend zieht er den Stift auf das gewünschte Menü (Quelle: [49]). (b) zeigt einen *Shadow Button*, welcher links oben an der Formel dargestellt ist. Hält der Nutzer den Stift über den grünen, transparenten Bereich, erscheinen die vier zusätzlichen Knöpfe (Quelle: nach [58]).

tion auf neue Programme anwenden. SBIs können unter Verwendung eines direkten Eingabegerätes dennoch natürlicher verwendet werden, da sich ihre Benutzung an einer Stift-und-Papier-Metapher orientiert, die vielen Menschen vertrauter ist.

FEHLERTOLERANZ Fehler bei der Eingabe können beim WIMP-Paradigma durch Drücken des falschen Symbols oder bei SBIs durch falsch gezeichnete oder interpretierte Gesten entstehen. Die Reaktion des Systems darauf ist wie bei der *Steuerbarkeit* stark von der konkreten Implementierung abhängig.

INDIVIDUALISIERBARKEIT Beim WIMP-Paradigma betrifft die mögliche Individualisierbarkeit hauptsächlich den Aufbau des UIs (Symbole und Menüs) und die Festlegung benutzerspezifischer Tastenkürzel. Bei SBIs betrifft dies die Abstimmung des Eingabegerätes auf den Nutzer oder das Einfügen von vom Nutzer definierter Gesten. Weiterhin kann die Gestenerkennung für individuelle Unterschiede der Nutzer beim Zeichnen verbessert werden.

LERNFÖRDERLICHKEIT Aussagekräftige Symbole und Menüs beim WIMP-Paradigma und zusätzliche Darstellung der möglichen Gesten können bei beiden Ansätzen den Nutzer anleiten und unterstützen und somit die Lernförderlichkeit erhöhen.

2.3.5.1 *Zusammenführung von WIMP und SBI*

Im Fokus dieser Arbeit stehen Ärzte bei der Patientenaufklärung. Ziel ist es daher, beide Ansätze (WIMP und SBI) so zu kombinieren, dass dem Arzt flexible Möglichkeiten für das Skizzieren von Gefäßen, deren Krankheiten und Behandlungsmöglichkeiten zur Verfügung gestellt wird *und* der Patient verfolgen und verstehen kann, was der Arzt tut. Für das Skizzieren eignet sich daher ein SBI-orientierter Ansatz. Beim Umschalten von Modi würde ein gestenbasierter Ansatz hingegen hinderlich sein:

Abbildung 18: Die Abbildung zeigt ein komplexeres *inplace Menü*. Nach Tippen auf den Stern erhält der Nutzer verschiedene Optionen. Während er den Stift auf der Oberfläche hält, fährt er auf das Menü „Item“ und spezifiziert anschließend, was er damit tun möchte. Indem er auf den „Move“-Befehl zieht, kann der Nutzer den Stern verschieben (Quelle: [29]).

Zum einen müsste der Arzt Gesten zur Steuerung lernen und zum anderen könnte der Patient womöglich nicht verfolgen, welche Aktionen der Arzt durchführt. Eine WIMP-basierte Steuerung durch Menüs und Symbole würde diese Probleme beheben.

Eine Kombination von Menüs nach dem WIMP-Paradigma und dem Ausführen einer Aktion *in* der Zeichenoberfläche stellen sog. *inplace Menüs* bzw. *marking Menüs* dar. Diese werden nur angezeigt, wenn der Nutzer diese benötigt, wodurch der Zeichenbereich nur bei deren Verwendung verdeckt wird. Solche Menüs werden in [49], [29] und [58] verwendet. In [49] werden radiale Menüs beschrieben, die am Stift-Interaktionspunkt erscheinen. Nach dem Erscheinen kann einer der Menüeinträge (die kreisförmig angeordnet sind) ausgewählt werden, indem der Stift in dessen Richtung bewegt wird (siehe Abb. 17(a)). Dieser Ansatz wird in [29] erweitert, indem nach der Auswahl eines Menüeintrags weitere Menüeinträge gezeigt werden. Während der gesamten Auswahlphase bleibt der Stift dabei auf der Oberfläche und verfeinert die Auswahl in den Untermenüs (siehe Abb. 18). In [58] werden sog. *Shadow Buttons* vorgestellt. Diese beschreiben eine transparente Fläche, die an Objekte auf der Zeichenoberfläche gekoppelt und immer sichtbar ist. Hält der Nutzer nun den Stift über diese Fläche, so werden ihm Elemente des WIMP-Paradigmas wie Knöpfe angezeigt, die er auswählen kann (siehe Abb. 17(b)).[26]

26 Voraussetzung hierfür ist die Erkennung des sog. *hover*-Zustands des Stifts. Dieser tritt auf, wenn der Stift nah über der Zeichenoberfläche schwebt, diese aber nicht berührt.

3

KONZEPT

Das Konzept wird in Anlehnung an die in Kapitel 1 beschriebenen *Ziele* und unter Berücksichtigung der in Kapitel 2.3.4 beschriebenen *Usability-Kriterien* entworfen. Die Ziele werden implizit beschrieben, während die Usability-Aspekte explizit an den entsprechenden Stellen kenntlich gemacht werden. Der Aufbau folgt dabei inhaltlich dem Grundlagenkapitel 2. Zunächst wird daher auf die Blutfluss-Simulation eingegangen und anschließend das Skizzieren erläutert. Darauf folgt eine Beschreibung des UIs.

3.1 BLUTFLUSS-SIMULATION UND -DARSTELLUNG

3.1.1 *Vorbetrachtung*

Eine Anforderung an die entstehende Anwendung ist Echtzeitfähigkeit.[1] Dies betrifft zum einen das Usability-Kriterium der *Aufgabenangemessenheit*, da mit einer schnell antwortenden Anwendung effizientes Arbeiten möglich ist. Zum anderen wirkt sich eine echtzeitfähige Anwendung auf die *User Experience* aus, da dies zur Nutzerakzeptanz beiträgt. Bei der *Simulation des Blutflusses* sind daher Vorbetrachtungen nötig. Die Simulationsgeschwindigkeit ist stark von der verwendeten Gittergröße abhängig. Eine hohe Gittergröße führt zu einer detailreicheren Simulation, benötigt allerdings auch mehr Zeit für die Berechnung. Die simulierte Flüssigkeit soll auf Hindernisse reagieren können (z. B. Gefäße). Im Gitter entspricht ein Hindernis einer blockierten Gitter-*Zelle*. Um dem Nutzer möglichst detaillierte Hindernisse zu ermöglichen, sollte die Gittergröße also möglichst hoch sein. Erste Implementierungen für diese Arbeit zeigten, dass die Berechnungen auf einer herkömmlichen Central Processing Unit (CPU) bei einer Gittergröße von über 100x100 zu einer Framerate von unter 15 FPS führten.[2] Da dies für den gewünschten Detailgrad der Hindernisse zu restriktiv ist, wird in dieser Arbeit für das Lösen der Berechnungen die GPU genutzt.[3] Laut [32] kann durch die parallele Verarbeitungsfähigkeit der GPU die sechsfache Geschwindigkeit bei der Berechnung erreicht werden. Die sich hieraus ergebende

1 Die Darstellung von Einzelbildern sollte schnell genug sein, sodass der Nutzer keine Einzelbilder, sondern eine flüssige Darstellung wahrnimmt. Ab 15 Frames pro Sekunde (FPS) kann eine Anwendung als echtzeitfähig angesehen werden. Dennoch sorgt eine Steigerung der FPS für flüssigere und ansprechendere Darstellung [2].

2 Verwendet wurde folgende CPU: Intel Core i5-2450M CPU @ 2,50GHz.

3 Die Verwendung der Grafikkarte für die Berechnungen, die über ihr ursprüngliches Einsatzgebiet hinaus gehen, wird als General Purpose Computation on Graphics Processing Unit (GPGPU)-Ansatz bezeichnet.

Notwendigkeit der Formulierung der Berechnung als bildbasiertes Problem wird im Implementierungskapitel 4 beschrieben.

Eine weitere Vorüberlegung betrifft die Repräsentation von Werten (wie Geschwindigkeit oder Druck) im Gitter. Einerseits können diese Werte im Zentrum jeder Zelle gespeichert werden. Dies wird *collocated* oder *zell-zentrierte* Diskretisierung genannt. Andererseits ist es möglich, die Werte in einem sog. *gestaffelten* (engl. *staggered*) Gitter zu speichern. Hierbei würden Werte (wie der Druck) weiterhin im Zentrum jeder Zelle gespeichert werden, die Geschwindigkeit hingegen *zwischen* den Zellen des Gitters. Diese Methode kann die Genauigkeit einiger Berechnungen erhöhen, ist bei der Umsetzung jedoch komplizierter, da sich verschiedene Gittergrößen ergeben [32].[4] Daher wird die zell-zentrierte Diskretisierung genutzt. Eine Beschreibung der Berechnungen für ein gestaffeltes Gitter ist in [9] zu finden.

3.1.2 *Verwendung der Navier-Stokes-Gleichungen*

Durch die im Grundlagenkapitel hergeleitete Berechnungsvorschrift für die Navier-Stokes-Gleichungen kann deren Berechnung in mehrere Schritte aufgeteilt werden. In Abbildung 19 sind die einzelnen Berechnungsschritte sowie ihre Auswirkung auf ein Geschwindigkeitsvektorfeld schematisch dargestellt. Die einzelnen Komponenten werden im Folgenden genauer erläutert.

3.1.2.1 *Externe Kräfte*

Das anfangs ruhende Geschwindigkeitsvektorfeld soll vom Nutzer während der Simulation manipuliert werden können. Hierdurch ist die Abbildung des Verhaltens eines *unstetigen Fluids* möglich. Die Manipulation und somit die Anwendung einer externen Kraft erfolgt durch Skizzierung, weshalb dies im Skizzierungskapitel 3.2.3 näher erläutert wird. Nachdem der Nutzer eine Kraft skizziert hat, die auf das Vektorfeld wirkt, übt diese *stetig* eine Veränderung des Geschwindigkeitsvektorfelds aus. Das gleichmäßige Einwirken von Kraft entspricht aber nicht dem Fließverhalten von Blut, da dieses durch das schlagende Herz regelmäßig beschleunigt wird. Um das Fließen des Bluts möglichst realitätsnah nachzuahmen, wird versucht, die einwirkende Kraft des Herzens mithilfe einer Funktion abzuschätzen:

$$p = f(x) = \begin{cases} 3 \cdot \sin x + 1, & x \in [3\pi \cdot k,\ (3k+1)\pi], k \in \mathbb{Z} \\ 1, & x \in ((3k+1)\pi,\ 3\pi \cdot k), k \in \mathbb{Z} \end{cases},$$

wobei p einen Faktor beschreibt, mit dem die vom Nutzer erzeugte Kraft multipliziert wird. Für x wird die Anzahl der berechneten Frames, geteilt durch 100 verwendet. Der Faktor p wiederholt sich also alle $3\pi * 100 \approx 943$ Frames. In Abbildung 20 ist die Funktion sowie die Auswirkung auf die Simulation dargestellt.

3.1.2.2 *Advektion*

Für die Advektion wird das Semi-Lagrange-Verfahren verwendet. Die im Grundlagenkapitel erwähnte Literatur diskutierte Integrationsverfahren und präferierte dabei die Runge-Kutta-Integration gegenüber der Euler-Integration, um ein Abdriften

4 Für ein 3x3-Gitter müssten für den Druck neun Werte gespeichert werden. Für die Geschwindigkeiten, die an jeder Kante des Gitters gespeichert werden, ergeben sich 24 Werte.

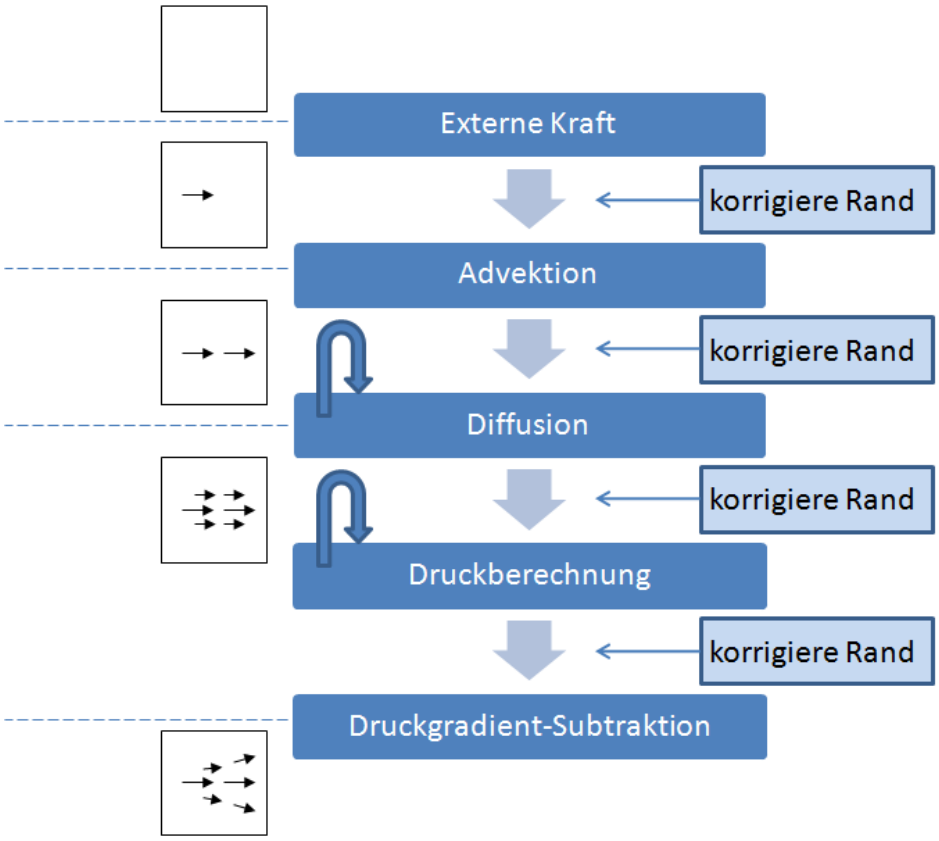
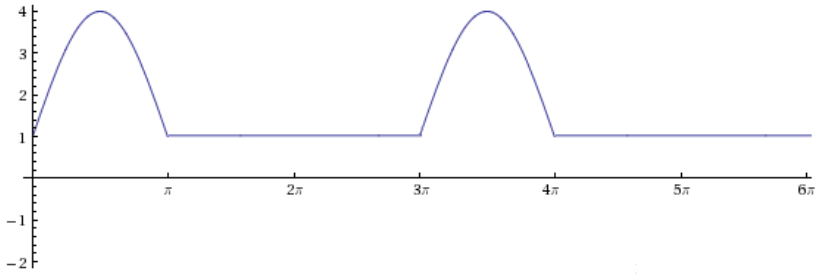
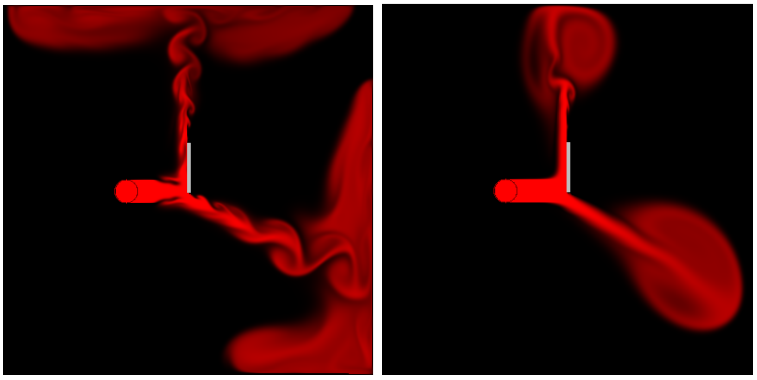


Abbildung 19: Darstellung der einzelnen Schritte für die Berechnung der Navier-Stokes-Gleichungen. Durch selbstverweisende Pfeile werden Schritte gekennzeichnet, die wiederholt durchlaufen werden, um das Simulationsergebnis zu verbessern. Zusätzlich wird schematisch die Auswirkung auf ein Geschwindigkeitsvektorfeld illustriert.

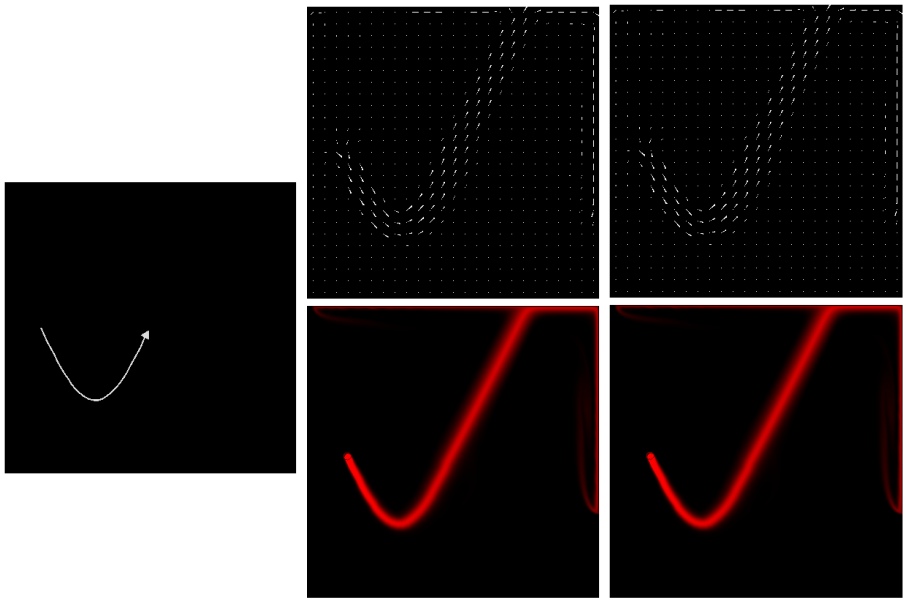


(a)

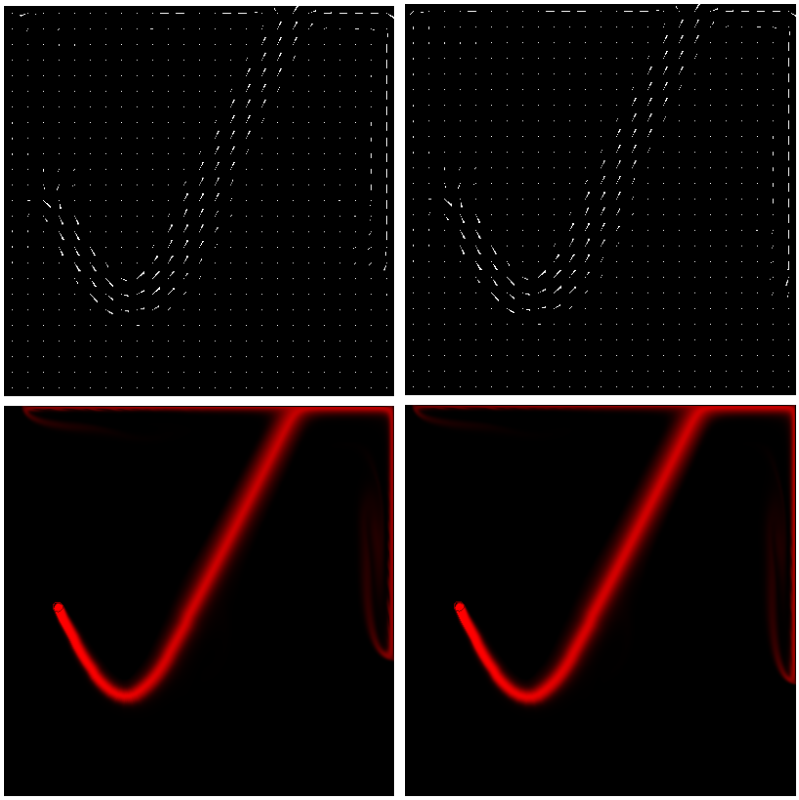


(b) (c)

Abbildung 20: In (a) ist die Funktion dargestellt, mit deren Ergebnis die externe Kraft multipliziert wird. In (b) ist der Blutfluss mit und in (c) ohne die pulsierende Kraft dargestellt (Frame: 800, Jacobi-Iterationen: 40, $\Delta t = 0,2$).



(a) Kraft (b) RK2 (c) RK3



(d) Euler (e) RK4

Abbildung 21: In (a) ist die Kraft illustriert, mit der das Geschwindigkeitsvektorfeld manipuliert wird. In (b)-(e) sind verschiedene Integrationsverfahren dargestellt, die jeweils durch zwei Visualisierungsverfahren repräsentiert werden, die in Abschnitt 3.1.3 näher erläutert werden. Der Vergleich von der Euler-Integration (d) und dem vierstufigen Runge-Kutta-Verfahren (e) zeigt, dass der Unterschied beider Verfahren für einen Integrationschritt von $\Delta t = 0,3$ minimal ist (Frame: 500, Jacobi-Iterationen: 40).

bei Vektorfeldern mit zu starker Krümmung zu vermeiden. Zu erwähnen ist hierbei die Integrationsschrittweite. Mit dieser lässt sich Einfluss darauf nehmen, wie weit von einem Punkt aus advektiert wird, indem die Integrationsrichtung mit ihr multipliziert wird. Aus diesem Parameter ergibt sich somit auch eine Steuervariable für die Geschwindigkeit der Simulation, weshalb sie im Folgenden mit Δt bezeichnet wird. In Abb. 21 sind die Euler-Integration sowie das zwei-, drei- und vier-stufige Runge-Kutta-Verfahren dargestellt. Interessant ist, dass sich trotz der aufwendigeren Berechnung beim Runge-Kutta-4-Verfahren kaum Unterschiede zur einfachen Euler-Integration ergeben. Dies resultiert aus der Simulationsgeschwindigkeit von $\Delta t = 0,3$, welche zu kleinen Advektionsschritten und somit ähnlichen Integrationsergebnissen bei den Euler- und Runge-Kutta-Verfahren führt. Aufgrund der geringen Unterschiede beider Verfahren wird die Euler-Integration verwendet, da sie performanter ist.

Nachdem mithilfe der Integration eine Position ermittelt wurde, muss, da sich die Position zumeist *zwischen* anderen Gitterpunkten befindet, aus den Nachbarn ein Wert berechnet werden. Dabei werden standardisierte Funktionen der GPU verwendet, da diese bereits performant implementiert sind. Für die Bestimmung des neuen Werts sind zwei Möglichkeiten anwendbar: Es kann der

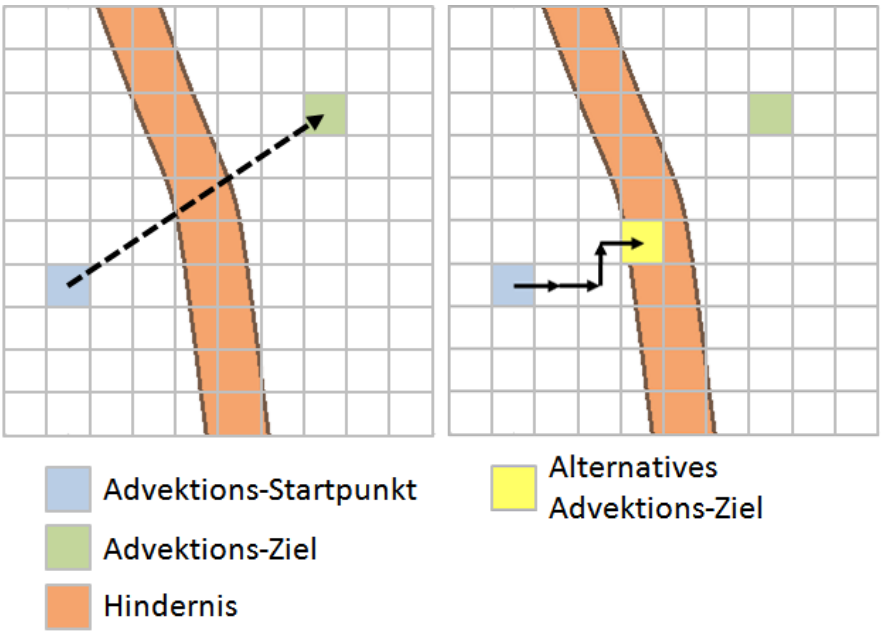
- nächste Nachbar oder
- bilineare Interpolation

genutzt werden. Aufgrund der besseren Eigenschaften der bilinearen Interpolation wird diese verwendet.

Ein weiterer Fall ist zu beachten: Wenn bei der Advektion zwischen dem Startpunkt und dem durch die Integration ermittelten Punkt ein Hindernis liegt, so dürfte die Geschwindigkeit von diesem nicht übernommen werden, da das Hindernis dies verhindern müsste. Dies könnte durch kleinere Integrationsschritte oder breitere Hindernisse vermieden werden, würde aber die Flexibilität der Anwendung einschränken. Stattdessen wird die Euler-Integration folgendermaßen erweitert: Vom Start- bis zum Endpunkt wird nicht gesprungen, sondern Gitterzelle für Gitterzelle *gewandert*. Somit wird jede Gitterzelle zwischen dem Start- und Endpunkt überprüft. Sofern ein Hindernis dazwischen liegt, wird die Geschwindigkeit des Hindernisses übernommen (siehe Abb. 22).

3.1.2.3 *Diffusion*

Wie im Grundlagenkapitel beschrieben ist für die Berechnung der Diffusion, also die Ausbreitung von Flüssigkeit oder Kräften im Vektorfeld, das Lösen eines linearen Gleichungssystems nötig. Ein Ergebnis kann durch die Verwendung eines *Solvers* abgeschätzt werden, welcher iterativ angewendet wird und bei jeder Iteration das Ergebnis verbessert. Während hierfür verschiedene Ansätze existieren, die mit den Eigenschaften Implementierungsaufwand, Berechnungsgeschwindigkeit und Konvergenzgeschwindigkeit unterschieden werden können, muss zusätzlich deren Umsetzbarkeit auf der GPU betrachtet werden. In [28] werden verschiedene Implementierungen auf der GPU beschrieben und hinsichtlich der Berechnungsgeschwindigkeit miteinander verglichen. Die dortigen zwei schnellsten Verfahren sind das Jacobi- sowie



(a) (b)

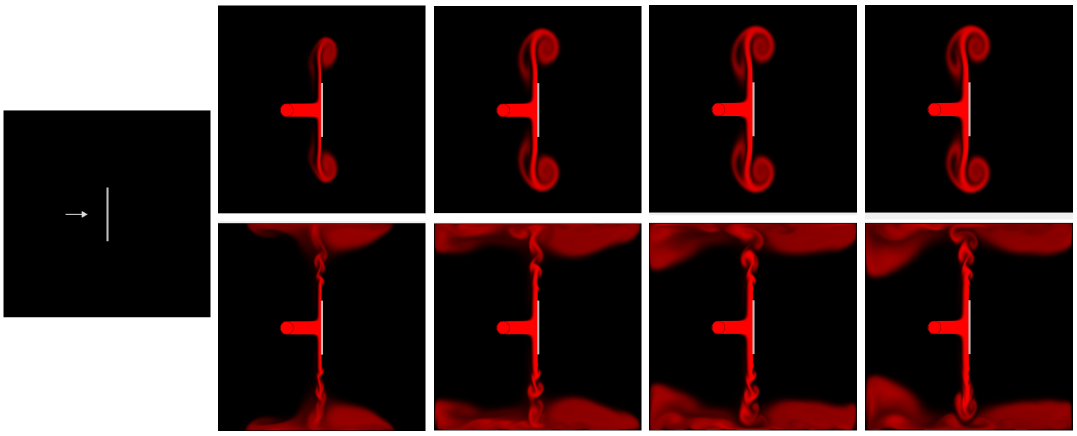
Abbildung 22: In (a) ist die normale Euler-Integration dargestellt. Bei der Berechnung des Advektionsziels wird das zwischenliegende Hindernis ignoriert. In (b) ist das angepasste Verfahren dargestellt. Die Zellen werden in Richtung Advektionsziel Schritt für Schritt abgewandert. Wenn dabei auf ein Hindernis gestoßen wird, wird diese Zelle als Advektionsziel betrachtet.

das Gauß-Seidel-Verfahren. Aufgrund der wesentlich aufwendigeren Implementierung des Gauß-Seidel-Verfahrens wird dem Vorbild von [32] gefolgt und das Jacobi-Verfahren verwendet. Dieses zeichnet sich durch eine einfache Implementierung aus und die nötigen Berechnungen können performant von der GPU ausgeführt werden. Anzumerken ist, dass auch bei ähnlicher Berechnungsgeschwindigkeit des Gauß-Seidel-Verfahrens dessen Konvergenzverhalten schneller ist und durch dessen Umsetzung ein Verbesserungspotential besteht. Ein Iterationsschritt des Jacobi-Verfahrens für eine Zelle lässt sich mithilfe der FDM durch folgende Formel beschreiben:

$$v_{i,j}^{(k+1)} = \frac{v_{i-1,j}^{k} + v_{i+1,j}^{k} + v_{i,j-1}^{k} + v_{i,j+1}^{k} + \alpha v_{i,j}}{\beta}.$$

v beschreibt das Geschwindigkeitsvektorfeld und die Indizes i und j beschreiben die Koordinate der ausgewählten Zelle. Durch α und β wird der Abstand des Gitters (1), die Viskosität ($visc$) und die Nachberschaftsanzahl (4) beschrieben. Hierdurch ergibt sich $\alpha = 1 \cdot \frac{1}{visc}$ und $\beta = 4 + \alpha$.

Die Verwendung von Diffusion sorgt für ein realistischeres Verhalten des Bluts. Weiterhin hängt von ihr die Simulation der Viskosität ab. Ein Nachteil der Diffusion ist allerdings, dass Details wie Verwirbelungen durch die entstehende Ausbreitung geglättet werden und somit nicht mehr sichtbar sind. Da für diese Arbeit ein realistisches Verhalten *und* Details wichtig sind, wird die Verwendung von Diffusion dem Nutzer als Option angeboten.



(a) (b) JI: 10 (c) JI: 30 (d) JI: 50 (e) JI: 70

Abbildung 23: In (a) sind die Kraft sowie ein Hindernis dargestellt. Die Darstellungen (b)-(e) zeigen verschiedene Anzahlen von Jacobi-Itertionen (JI) im Simulationsframe 500 (oben) und 1500 (unten) ($\Delta t = 0,2$).

3.1.2.4 *Berechnung des Drucks*

Die Herleitung für die Berechnung des Drucks wurde in Kapitel 2.2.1.1 beschrieben und dient der Einhaltung der zweiten Navier-Stokes-Gleichung. Diese sichert die Inkompressibilität des Geschwindigkeitsvektorfeldes, indem festgelegt wird, dass es *divergenzfrei* ist. Das durch die vorherigen Operationen (externe Kräfte, Advektion und Diffusion) erhaltene Geschwindigkeitsvektorfeld erfüllt diese Bedingung nicht, weshalb es korrigiert werden muss. Dies geschieht, indem der Gradient des Druck-Vektorfeldes mithilfe eines linearen Gleichungssystems bestimmt wird (siehe Kapitel 2.2.1.1). Auch hierfür wird wie bei der Diffusion, das Jacobi-Verfahren genutzt. Die Formel unterscheidet sich leicht von der bei der Diffusion verwendeten:

$$p_{i,j}^{(k+1)} = \frac{p_{i-1,j}^{k} + p_{i+1,j}^{k} + p_{i,j-1}^{k} + p_{i,j+1}^{k} + \alpha d_{i,j}}{\beta}.$$

Hierbei ist p das Skalarfeld, i und j sind die Indizes der jeweiligen Zelle und d ist ein Skalarfeld, welches die Divergenz von dem Geschwindigkeitsvektorfeld u ($d = \nabla \cdot u$) enthält. α und β beschreiben den Gitterabstand ($\alpha = 1$) sowie die Nachbarschaftsanzahl ($\beta = 4$).

Die Anzahl der Iterationsschritte entscheidet hierbei über die Qualität der Fluss-Simulation. Eine geringe Anzahl führt zu einem unnatürlichen Verhalten der Flüssigkeit, da kaum Verwirbelungen entstehen. Dies wirkt sich wiederum negativ auf die *Erwartungskonformität* aus. Eine hohe Anzahl bedeutet einen hohen Berechnungsaufwand. In Abbildung 23 sind verschiedene Iterationsschritte und deren Auswirkungen dargestellt. Ab 30 Iterationsschritten ergeben sich visuell ansprechende Ergebnisse, weshalb dies die Grundeinstellung der Anwendung ist. Dem Nutzer wird zusätzlich ermöglicht, die Iterationsanzahl während der Nutzung des Programms zu verändern.

3.1.2.5 *Randwertbedingungen*

Randwertbedingungen sind zur Anwendung der im Grundlagenkapitel beschriebenen FDM nötig. Während diese in Fluid-Simulationen üblicherweise genutzt werden, um korrektes Flussverhalten am Rand des Gitters zu gewährleisten (Flüssigkeit darf nicht austreten), können sie auch genutzt werden, um beliebige Hindernisse im Gitter zu platzieren. Randwertbedingungen müssen zum einen für das Geschwindigkeitsvektorfeld und zum anderen für das Skalarfeld des Drucks gewährleistet werden. Für das Geschwindigkeitsvektorfeld wird eine spezielle Dirichlet-Randwertbedingung genutzt, die *no-slip*-Bedingung. Diese sorgt dafür, dass die Flüssigkeit am Rand nicht austreten kann und sie vom Rand *festgehalten* wird. Um dies zu erreichen muss sichergestellt werden, dass die Geschwindigkeit relativ zum Hindernis Null ist. Hierbei reicht es nicht aus, die Geschwindigkeit im Hindernis auf Null zu setzen. Da es sich bei der Gitterrepräsentation um die zell-zentrierte Diskretisierung handelt (siehe Vorbetrachtungsabschnitt 3.1.1), muss die Geschwindigkeit im Hindernis so gewählt werden, dass sie an der Zellenkante, also *zwischen* der Hinderniszelle und der Flüssigkeitszelle, auf Null abfällt. Dies ist durch folgende Berechnung möglich:

$$\frac{v_{Hindernis} + v_{Nachbar}}{2} = 0,$$

wobei v der Geschwindigkeitsvektor der jeweiligen Zelle ist. Die Teilung durch 2 ist nötig, um den Wert *zwischen* den Zellen zu beschreiben. Nach Umstellung erhält man die Vorschrift, mit der der Wert für das Hindernis berechnet werden kann: $v_{Hindernis} = -v_{Nachbar}$.

Für beliebige (z. B. runde) Hindernisse muss für die Berechnung der relevante Nachbar bestimmt werden. Dieser kann mithilfe der Normale an einer Hindernis-Zelle gefunden werden. Aufgrund der Diskretisierung des Gitters muss die Normale abgeschätzt werden. Hierbei wird dem Ansatz von [100] gefolgt und die Normale mithilfe der Betrachtung der Einer-Nachbarschaft abgeschätzt (siehe Abb. 24). Um die Genauigkeit der Abschätzung zu erhöhen könnte eine größere Nachbarschaft betrachtet werden. Dies würde allerdings sowohl zu einer aufwendigeren Berechnung führen, als auch die Komplexität der Unterscheidung erhöhen.

Wie im Grundlagenkapitel beschrieben, müssen die Randwertbedingungen auch für das Skalarfeld des Drucks durch die Neumann-Randbedingung eingehalten werden. Dies wird durch folgende Formel gewährleistet:

$$\frac{\partial p}{\partial n} = 0.$$

Die Ableitung des Drucks in Richtung der Normale wird Null. wobei ∂p die Ableitung der Drucks in einer Zelle ist und ∂n die Ableitung der Normale. Mithilfe der Vorwärts-Finite-Differenzen-Methode lässt sich ∂p umformen und es ergibt sich:

$$\frac{p_{Nachbar} - p_{Hindernis}}{\partial n} = 0.$$

Die Gleichung kann also erfüllt werden, indem $p_{Nachbar} = p_{Hindernis}$ gesetzt wird.

Die beschriebenen Korrekturen der Randwertbedingung sind nach jeder Veränderung des Geschwindigkeitsvektorfelds und des Druck-Skalarfelds nötig (siehe Abb. 19).

Abbildung 24: Dargestellt sind Zellen die als *Flüssigkeit* oder als *Hindernis* gekennzeichnet sind. Durch die Betrachtung der Einer-Nachbarschaft der Hindernis-Zellen kann deren Normale abgeschätzt werden (Quelle: nach [100]).

3.1.3 *Darstellung des Blutflusses*

Nachdem das Geschwindigkeitsvektorfeld berechnet wurde, muss eine geeignete Methode gefunden werden, dieses darzustellen. Hierbei sind die Usability-Aspekte *Aufgabenangemessenheit* und *Erwartungskonformität* relevant, da die Visualisierung auf verständliche Weise einem Arzt und Patienten kommunizieren muss, wie sich der Blutfluss verhält. Eine Methode zur Darstellung aus der Klasse der *direkten Strömungsvisualisierungen* sind Linien- oder Pfeilplots. Aufgrund der Umsetzung auf der GPU ist das Platzieren von beliebigen Glyphen wie Pfeilen nicht ohne Weiteres möglich. Linienplots hingegen können recht einfach durch die GPU dargestellt werden, bieten aber keine Richtungsinformationen. Daher wird eine neue Visualisierungstechnik genutzt, die im Folgenden *Fächerplots* genannt wird. Die Grundidee besteht darin, die Ausrichtung eines Vektors mit Polarkoordinaten zu beschreiben. Somit besitzt dieser statt einer x- und y-Komponente nun einen Radius und einen Winkel. Während mit dem Radius die Stärke des Vektors kodiert wird, dient der Winkel dazu, die Richtung darzustellen. Würde man nun den exakten Winkel darstellen, so würde man einen Linienplot erhalten. Stellt man den Winkel hingegen mit einem Toleranzbereich dar, so entsteht ein *Fächer*, welcher in Richtung des Vektors breiter wird (siehe Abb. 25). Dieses Verfahren ist von der GPU ähnlich schnell zu berechnen wie der Linienplot. Zu diskutieren ist, ob die Verbreiterung in Richtung des Vektors beim Nutzer genau zur gegenteiligen Richtungsinterpretation führt. Während dies bei der Deutung des Fächers als Fahne, die sich im Wind bewegt, auftreten könnte, wird durch die dynamische Bewegung des Fächers während der Simulation die Richtung deutlich.

Die Fächerplots werden in einem regelmäßigen Gitter über der FlussS̄imulation platziert. Mit dem Abstand zweier Zellen des Gitters kann dabei Einfluss darauf genommen werden, wie viele Informationen des Strömungsfelds gezeigt werden sowie die kognitive Last des Nutzers erhöht wird. Neben der Betrachtung des Abstands muss festgelegt werden, welche Informationen durch einen Fächer repräsentiert werden. Zum einen kann die Ausprägung des Vektors dargestellt werden, an welchem sich der Fächer befindet, andererseits können die Vektoren der umliegenden Umgebung zusammengefasst werden. Während die erste Methode schneller zu berechnen

ist, besteht das Risiko, dass wichtige Informationen des Strömungsfelds nicht dargestellt werden. Daher wird die zweite Methode angewendet. Durch die Mittelung aller Nachbarvektoren wird die Richtung und Stärke des Fächers bestimmt. Zu beachten ist, dass durch die Glättung dennoch interessante Stellen des Strömungfelds übersehen werden können.

Diese Methode der Darstellung ist für die Patientenaufklärung eventuell ungeeignet, da sie das gesamte Strömungsfeld darstellt, anstatt nur relevante Bereiche hervorzuheben. Daher ist die Fächerplotdarstellung in der Grundeinstellung der Anwendung deaktiviert und kann bei Bedarf zugeschaltet werden.

Eine weitere Methode, die den Fokus auf vom Nutzer festgelegte Bereiche der Strömung lenkt, wird im Folgenden beschrieben. Das Augenmerk dieser Methode liegt darauf, eine ansprechende und leicht verständliche Visualisierung zu bieten, um dem Kriterium der *User Experience* zu entsprechen. Die Grundidee dieser Skalarfelddarstellung basiert auf der Analogie, *farbige Tinte* in Flüssigkeit zu platzieren. Mit der Menge an platzierter Tinte lässt sich der Wert im Skalarfeld steuern. Diese Werte können wie die Geschwindigkeitsvektoren durch Diffusion und Advektion bewegt werden. Die Darstellung des Wertes geschieht nun über eine geeignete Farbskala. Hierfür wird keine mehrfarbige Skala verwendet, um weitere Farben für zusätzliche Skalarfelder zu erhalten. Der Verlauf für ein Skalarfeld wird daher über eine Farbe definiert, die mit Abnahme des skalaren Wertes in Schwarz übergeht. Somit ist es möglich, an mehreren Stellen der Simulation Farbe zu platzieren, die sich vermischen kann. Hierdurch ist folgendes Szenario durch die Anwendung darstellbar:

Der Arzt kann dem Patienten das Flussverhalten von Blut in einem Aneurysma illustrieren. Anschließend wird durch eine geeignete Behandlung, z. B. einen Stent, der Blutfluss im Aneurysma abgeschwächt. Durch die Nutzung einer zweiten Farbe kann der Arzt dem Patienten zeigen, wie stark das Blut nun in das Aneurysma eintritt und wie stark es sich mit dem noch im Aneurysma befindlichen Blut vermischt.

Für die Verwendung mehrerer Farben müssen geeignete Farben identifiziert werden. Wie im Grundlagenkapitel erläutert, würde sich die Regenbogenfarbskala nur bedingt eignen. Zwar stellen die verschiedenen Farben einen nominalen Datentyp dar, weswegen die Farben keine inhärente Ordnung besitzen müssen, allerdings besitzen die Farben der Regenbogenskala keine gleichen Abstände bezogen auf die Hellig-

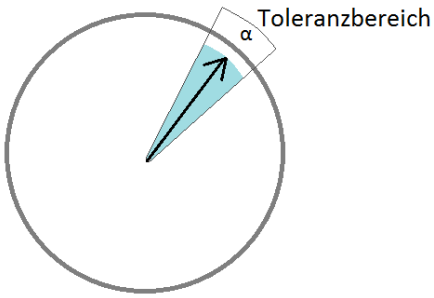
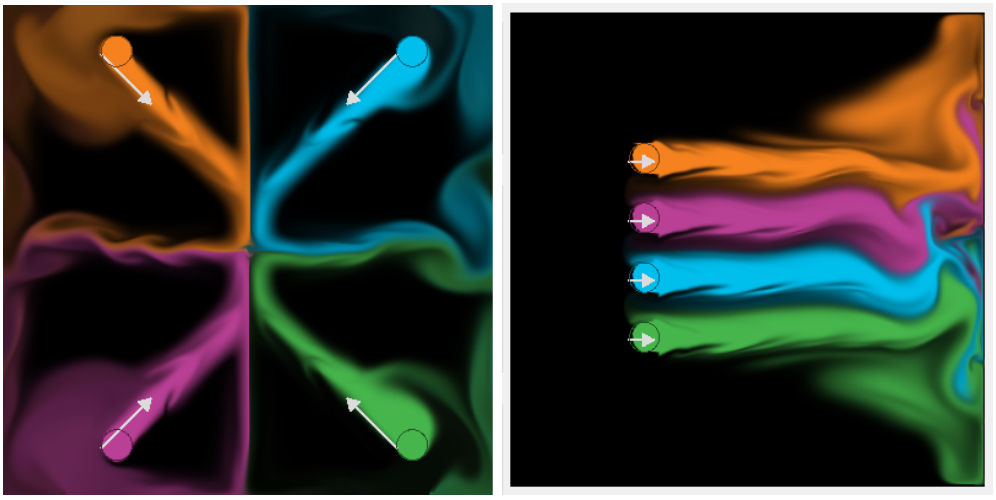


Abbildung 25: In schwarz ist der eigentliche Vektor dargestellt. Durch Betrachtung des Vektors in Polarkoordinaten kann dieser mit einem Radius und einem Winkel interpretiert werden. Stellt man nun den Winkel mit einem Toleranzbereich dar, so ergibt sich ein Fächer, der in Richtung des Vektors breiter wird.



(a) (b)

Abbildung 26: In den Darstellungen (a) und (b) werden jeweils vier Farben aus dem Lab-Farbraum verwendet.

keit. Für die Auswahl der Farbe wird daher auf den sog. *Lab-Farbraum* zurückgegriffen.[5] Dieser erlaubt eine einfache Bestimmung von Farben gleicher Helligkeit. Um zusätzlich eine gleichmäßige Unterscheidung des Farbtons zu erreichen, wird die Methode von [27] verwendet. Dort wird ein Kreis im Lab-Farbraum platziert und durch Abschreitung von Winkeln fester Länge die Parameter für a und b bestimmt. Es wird bei einem Orange begonnen ($L = 67, a = 43, b = 74$). Die Schrittweite richtet sich nach der gewünschten Anzahl von Farben. Für die vorliegende Arbeit werden vier verschiedene Farben genutzt (siehe 26). Falls die Farben nicht ausreichen könnte dem Nutzer angeboten werden, selbst eine Farbe aus dem Lab- oder Rot, Grün, Blau (RGB)-Farbraum zu wählen.

Wenn das Verfahren erweitert wird, indem stetig Tinte am selben Ort hinzugefügt wird, entsteht eine Strömungsvisualisierung aus dem Bereich der *stellenweisen Darstellung von Partikelverfolgung* (siehe Abschnitt 2.2.2.2): eine *Streakline*. Zwar werden keine Partikel verbunden, die kontinuierlich an einer Stelle über eine Zeitspanne losgelassen werden, dennoch ergibt sich eine zusammenhängende Darstellung. Das Finden eines geeigneten Saatpunktes wird dem Nutzer überlassen. Hierdurch kann er flexibel bestimmen, welche Bereiche er durch die beschriebene Technik darstellen will.

Eine weitere Technik für die Darstellung des Blutflusses könnte aus der Klasse der *dichten Darstellung von Partikelverfolgung* entnommen werden (siehe 2.2.2.2). Um das unstetige, zeitveränderliche Vektorfeld darzustellen, könnte die DLIC-Technik genutzt werden [88]. Diese kann das gesamte Strömungsfeld und wichtige Details illustrieren. Da diese Technik primär von Strömungsfeld-Experten verwendet wird, ist es möglich, dass sie für den Patienten oder den Arzt nicht zielführend ist. Daher wird sie nur konzeptionell erwähnt und nicht umgesetzt.

5 Der Lab-Farbraum ist in der Norm EN ISO 11664-4 beschrieben. Das Wort *Lab* ergibt sich aus den drei Komponenten, aus denen die Farbe zusammengesetzt wird. Mit *L* (lightness) wird die Helligkeit beschrieben, mit *a* ein Verlauf von *Grün zu Rot* und mit *b* ein Verlauf von *Gelb zu Blau*.

3.2 SKIZZIEREN

Für das Skizzieren der Gefäße, Gefäßerkrankungen, Behandlungsmöglichkeiten und des Blutflusses sind mehrere Usability-Kriterien relevant. Die *Selbstbeschreibungsfähigkeit* wird adressiert, da sie dem Nutzer kommuniziert, wie selbsterklärend er nötige Werkzeuge der Anwendung nutzen kann. Die *Aufgabenangemessenheit* ist relevant, da sie ein Qualitätsmaß für Bedienung der Werkzeuge darstellt. Für eine skizzenbasierte Anwendung kann *Erwartungskonformität* erreicht werden, indem der Interaktionsstil konsistent einer Papier-Stift-Metapher verfolgt. Mit der *Steuerbarkeit* wird gewährleistet, dass der Nutzer Kontrolle über die Abläufe (z. B. Reihenfolge der Benutzung der Werkzeuge) erhält und fehlerhafte Eingaben korrigieren kann.

3.2.1 *Verarbeitung der Eingabe*

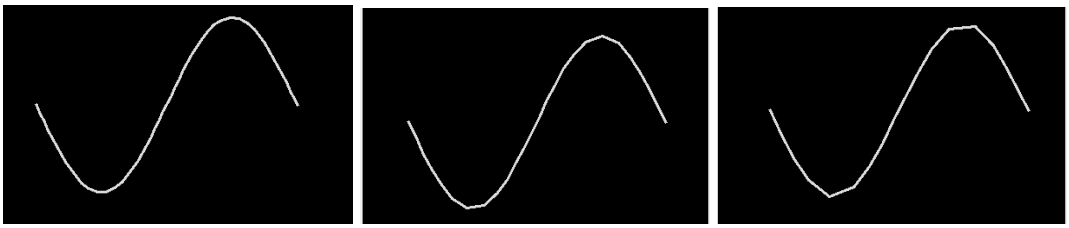
Die Anwendung soll möglichst flexibel bezogen auf das verwendete direkte Zeigegerät sein. Das heißt, dass sowohl die Stift- als auch die Touch-Eingabe unterstützt werden soll. Hieraus ergeben sich Einschränkungen:

- Spezielle Funktionen von Stift-Eingabegeräten, wie Drucksensitivität oder zusätzliche Knöpfe, werden nicht unterstützt.
- Die Multitouch-Eingabe wird nicht unterstützt.

Durch diese Einschränkungen ist das Konzept theoretisch auch unter Verwendung der linken Maustaste anwendbar.

Wie im Grundlagenkapitel beschrieben, sind die vom Eingabegerät erhaltenen Daten verrauscht. Für die Weiterverarbeitung ist es daher nötig, die Daten zu entrauschen und dabei möglichst wenig und gleichmäßig verteilte Positionsinformationen zu erlangen. Durch Verfahren wie *Resampling* und *Glättung* kann dies erreicht werden. Diese Verfahren werden häufig nach dem Zeichnen (also nach dem Absetzen des Stiftes) angewendet, was zu einer plötzlichen Veränderung des Gezeichneten führt. In dieser Arbeit werden Methoden vorgestellt, die *on-the-fly* (also während des Zeichnens) angewendet werden, wodurch die plötzliche Veränderung des Gezeichneten vermieden werden kann. Daraus resultiert vorhersagbares Verhalten der Anwendung, es führt aber auch zur Aufgabenstellung, echtzeitfähige Methoden zu erstellen, die eine flüssige Interaktion nicht einschränken.

Zunächst wird durch ein *Resampling* die Anzahl der Eingabepunkte reduziert. Ein simpler Ansatz hierfür wäre nur jeden n-ten erfassten Punkt zu verarbeiten und die restlichen Punkte zu verwerfen. Dieser Ansatz ist problematisch, da die verworfenen Punkte Strukturen betreffen könnten, die vom Nutzer gewünscht sind. Weiterhin führt er zu einem uneinheitlichen Abstand der Punkte, der aus der variierenden Zeichengeschwindigkeit des Nutzers resultiert. Stattdessen wird eine Methode verwendet, die zu einer gleichmäßigen Verteilung der Positionen führt: Es werden nur Punkte aufgenommen, die eine bestimmte Distanz d zum vorherigem Punkt haben, der Rest wird verworfen. Dieser Ansatz führt dazu, dass kurze Strecken bestehend aus Zickzacklinien nicht gezeichnet werden können. Dieser Umstand kann vernachlässigt werden, da die zu zeichnenden Strukturen (z. B. Gefäße oder Coils) keine zickzack-förmigen Elemente enthalten. Durch den Distanzparameter kann Einfluss auf die resultierenden Linienzüge genommen werden. In Abhängigkeit vom zu



(a) d = 10 (b) d = 20 (c) d = 30

Abbildung 27: In den Abbildungen (a)-(c) sind verschiedene Größen für den Distanzparameter *d* dargestellt.

zeichnenden Objekt kann mit ihm ein filigranes Zeichnen von Details ermöglicht werden (für z. B. Coils) oder die Details für breitere Strukturen vernachlässigt werden (für z. B. Stents). In Abbildung 27 sind verschiedene Werte für den Distanzparameter *d* dargestellt. Wie im Grundlagenabschnitt 2.3.3 erläutert, ist für das Resampling auch die Verwendung eines komplexeren Verfahrens möglich. Da die beschriebene, vergleichsweise einfache Methode aber *visuell befriedigende Ergebnisse* liefert und sie *on-the-fly* berechnet werden kann, wird diese verwendet.

Um den erhaltenen Linienzug zu glätten, wird ein lokaler Gauß-Filter (ähnlich zu [91]) auf die Punkte angewendet. Dieser betrachtet jeweils einen Punkt in einer Einer-Nachbarschaft und mittelt diesen auf Basis einer Gauß-Funktion. Der Filter wird dabei nicht jedes Mal auf alle Punkte angewendet (dies würde zu einem *Schrumpfen* des Linienzuges führen, welches bei längeren Linienzügen zunimmt). Stattdessen wird der Filter auf den *vorletzten* Punkt angewendet, nachdem ein neuer hinzugefügt wurde.

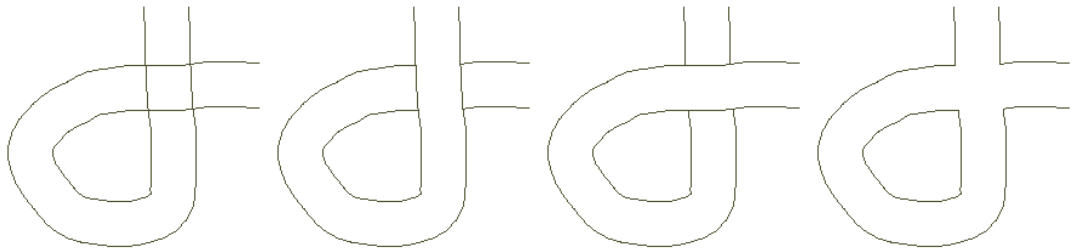
Weitere, berechnungsaufwendigere Techniken, um die Eingabedaten als Linienzug zu interpretieren, sind z.B Beziérkurven, B-Splines oder stückweise lineare Approximation. Da das Ergebnis, das durch das Resampling und den Gauß-Filter erhalten wurde, den Anforderungen genügt und nicht berechnungsaufwendig ist, werden diese nicht weiter betrachtet.

3.2.2 *Gefäße- und Gefäßerkrankungen*

Zum Zeichnen eines Gefäßes können zwei grundlegende Ansätze verfolgt werden:

- Der Arzt zeichnet zwei Linien, um das Gefäß zu beschreiben.
- Der Arzt zeichnet eine Linie und die Anwendung generiert daraus ein Gefäß.

Der erste Ansatz wird aus mehreren Gründen nicht weiter verfolgt: Das Erstellen von Gefäßen mit zwei Linien ist nicht nur aufwendiger, sondern erschwert auch eine gleichmäßige Skizzierung eines Gefäßes. Versucht der Nutzer eine zweite Linie zu skizzieren die parallel zur ersten verläuft, so muss er sehr genau zeichnen. Skizzierte Unregelmäßigkeiten können sich dabei auf die Blutfluss-Simulation auswirken. Dennoch sei erwähnt, dass der erste Ansatz zur Generierung von Gefäßen unterschiedlicher Dicke (z. B. Arteriolen und Kapillaren) geeigneter wäre. Im Folgenden wird die zweite Variante beschrieben, bei welcher der Nutzer unter Verwendung eines *Gefäß*-Werkzeugs ein Gefäß erstellt. Während der Nutzer eine Linie zeichnet, sollte



(a) (b) (c) (d)

Abbildung 28: In den Abbildungen sind vier Varianten dargestellt, wie sich überschneidende Gefäßstrukturen vom System interpretiert werden können.

ihm dabei das Gefäß dargestellt werden, welches aus der Linie resultiert. Hierdurch kann er antizipieren, wie die Umwandlung einer Linie in ein Gefäß funktioniert. Ein relevanter Parameter ist die Gefäßbreite. Nach der Überlegung, ob dieser Parameter einstellbar sein sollte, wurde sich für die Festlegung einer konstanten Breite entschieden. Dies resultiert aus den folgenden betrachteten Möglichkeiten, die Breite einzustellen:

- Der Nutzer könnte die Breite über ein (textuelles) Eingabefeld festlegen. Diese Methode ist bei stiftbasierter Interaktion nur bedingt geeignet.
- Der Nutzer könnte die Breite durch eine Geste, z. B. die Länge eines gezeichneten Strichs, festlegen. Dies würde zu einer komplexeren Zeicheninteraktion führen.

Die Festlegung der Breite schränkt zwar die Flexibilität der Anwendung ein, vereinfacht sie aber auch.

In der Arbeit von [101] wird die Zeichenfläche als 2,5D-Raum betrachtet. Hierdurch ist es möglich Gefäßverläufe entlang über und unter anderen Gefäßen zu skizzieren. In der vorliegenden Arbeit ist die Zeichenfläche hingegen zweidimensional, weswegen solche Verläufe nicht abbildbar sind. Daher stellt sich die Frage, was ein Nutzer erwartet, wenn sich zwei Linienzüge (und somit Gefäße) kreuzen. Folgende vier Möglichkeiten sind denkbar und in Abbildung 28 illustriert:

1. Die Gefäßwand *kreuzt* die Vorherige.
2. Das Gefäß verläuft *unter* dem Vorherigen.
3. Das Gefäß verläuft *über* dem Vorherigen.
4. Die Gefäße werden *ohne Kreuzung* zusammengefügt.

Von diesen Möglichkeiten wird die Vierte ausgewählt. Gegen die erste Variante ist einzuwenden, dass durch die Kreuzung das Gefäß blockiert wird, was nicht in der Absicht des Nutzers liegt. Die zweite und dritte Möglichkeit muss durch eine zusätzliche Eingabe oder durch Zeichenrichtung unterschieden werden. Ist für den Nutzer das Gefäß in der anderen Zeichenrichtung einfacher zu zeichnen, wird er dadurch eingeschränkt. Ein weiteres Problem der ersten drei Varianten ergibt sich durch die

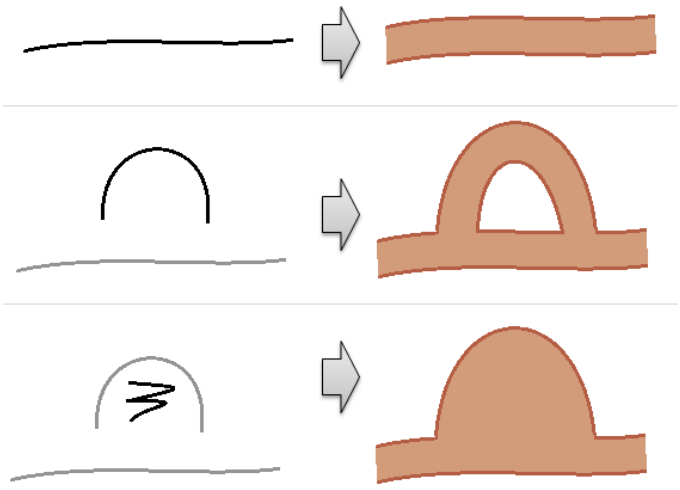
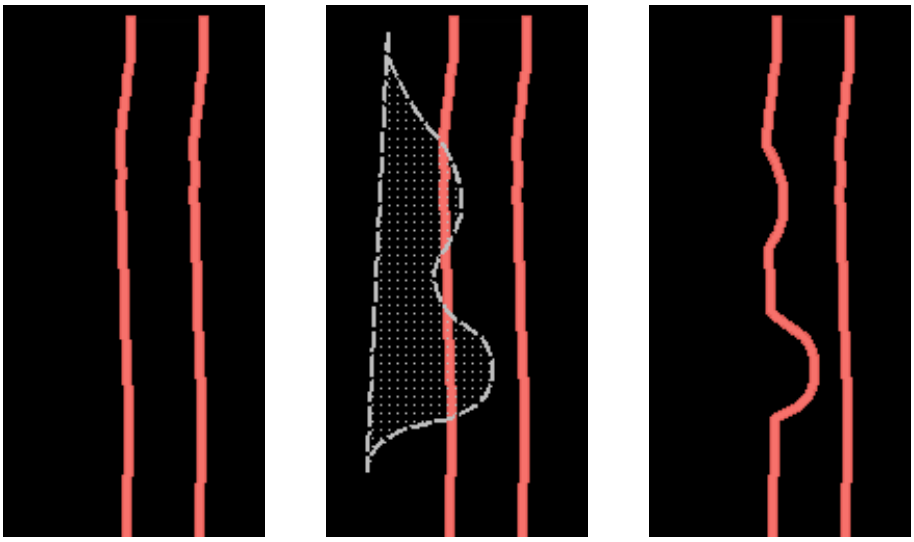


Abbildung 29: In der Abbildung sind jeweils die gezeichnete Linie (links) und das resultierende Gefäß dargestellt (rechts).

gewählte Implementierung. Kreuzen sich Gefäßwände so stellt dies für die Blutfluss-Simulation ein Hindernis dar und der Blutfluss würde an dieser Stelle stoppen, was nicht in der Absicht des Nutzers liegt.

Die Verwendung der vierten Möglichkeit hingegen bietet einen entscheidenden Vorteil. Durch die Vereinigung von überzeichneten Gefäßen hat der Nutzer die Möglichkeit, auf einfache Weise komplexe Gefäßstrukturen zu zeichnen. Gefäßabzweigungen können so durch Ansetzen an einem vorhandenen Gefäß und Neuzeichnen erzeugt werden. Weiterhin lassen sich hierdurch Aneurysmen realisieren. Der Nutzer *malt* einen Bereich aus, und beschreibt damit die Struktur des Aneurysmas (siehe Abb. 29).

Neben Aneurysmen soll die Anwendung das Zeichnen von Stenosen ermöglichen. Erste Überlegungen führten zur Idee eines *Zusammenzieh*-Werkzeugs. Dieses könnte auf einem bereits gezeichneten Gefäß angewendet werden; je länger der Nutzer dieses über einem Gefäß nutzt, desto weiter zieht es sich von beiden Seiten zusammen. Problematisch an diesem Ansatz ist, dass Stenosen nicht zwingend symmetrisch in einem Gefäß auftreten. Eine Alternative wäre (wie in [101]) die direkte Manipulation des Gefäßes zu ermöglichen, indem der Nutzer die Gefäßwand berührt und anschließend in die gewünschte Richtung zieht. Während die direkte Manipulation eine verständliche und leicht zu erlernende Möglichkeit ist Objekte zu manipulieren, steht sie im Kontrast zum Skizzierungsansatz der Anwendung, welcher an einer Stift-und-Papier-Metapher angelehnt ist. Daher wird stattdessen ein Ansatz gewählt, der dem Usability-Kriterium der Erwartungskonformität folgt und dem Nutzer eine konsistente, zeichenbasierte Funktion zum Erzeugen von Stenosen bietet. Dem Nutzer wird ein *Ausschneide*-Werkzeug angeboten, mit welchem er Elemente eines Gefäßes wegschneiden kann. In der Arbeit von Heckel et al. [34] wird gezeigt, dass Nutzer eine Schneide-Funktionen verschieden nutzen, wodurch eine mehrdeutige Interpretation der Nutzerintention möglich ist. Um solche Probleme mit dem *Ausschneide*-Werkzeug zu vermeiden, wird nicht nur die gezeichnete Kontur zum



(a) (b) (c)

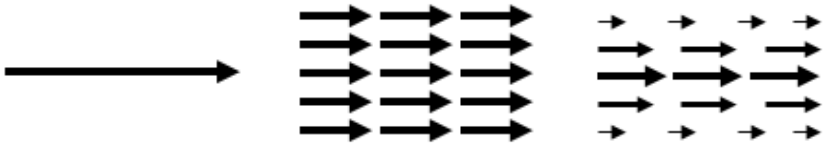
Abbildung 30: In (a) ist ein Gefäß dargestellt, welches durch die Verwendung des *Ausschneide*-Werkzeugs bearbeitet werden soll. Während der Bearbeitung wird der Schnittbereich durch die Kontur sowie durch die Verbindung von Start- und Endpunkt visuell hervorgehoben (b). Das Ergebnis der Bearbeitung ist in (c) illustriert.

Schneiden genutzt, sondern durch Verbindung der Kontur mit dem Start- und Endpunkt des Linienzuges eine Region aufgespannt (siehe Abb. 30). Diese Region wird beim Zeichnen visuell hervorgehoben und nach Absetzen des Stiftes vom bestehenden Gefäß abgezogen. Durch die Verwendung dieses Werkzeugs ist nicht nur das Erstellen von Stenosen möglich. In Kombination mit dem *Gefäß*-Werkzeug wird ein generisches Zeichenwerkzeug geschaffen, welches das Erstellen beliebiger Strukturen ermöglicht.

Eine Funktion, um den Arzt beim Zeichnen zu unterstützen, ist das Laden von Hintergrundbildern. Diese können dem Arzt als Vorlage und Hilfestellung zum Nachzeichnen dienen. Dabei kann der Arzt sein Wissen über Gefäße einfließen lassen und nur relevante Gefäßstrukturen nachzeichnen. Der Nachzeichnungsprozess kann unterstützt werden, indem sein gezeichnetes Gefäß *magnetisch* von den Konturen des Hintergrundbilds angezogen wird (siehe [68, 93]). Weiterhin ist zum einen das Laden eines patientenspezifischen Datensatzes oder die Verwendung von häufig verwendeten Gefäßstrukturen möglich, wie z. B. der *Circle of Willis*.[6]

Eine weitere nützliche Funktion würde die Möglichkeit darstellen, aus bildgebenden Verfahren gewonnene Gefäßstrukturen zu laden und diese direkt für die Blutfluss-Simulation zu nutzen. Hierbei entstehen mehrere Herausforderungen. Die Bilder enthalten neben Arterien auch Kapillaren. Durch Rauschen bzw. Artefakte durch die Bildgebung ist eine zuverlässige Identifizierung von Gefäßen schwierig. Weiterhin stellt ein Bild eine zweidimensionale Abbildung dar, wodurch Gefäße im Hinter- oder Vordergrund räumlich falsch auf einer Ebene dargestellt werden. Die-

6 Dieser nach Thomas Willis benannter Arterienring illustriert die Blutversorgung des Gehirns.



(a) (b) (c)

Abbildung 31: In (a) ist die Kraft abgebildet, wie sie dem Nutzer dargestellt wird. (b) zeigt, wie die Kraft als Region auf das Geschwindigkeitsvektorfeld gewirkt wird. In (c) wird die Kraft am Rand mit einer Gauß-Funktion abgeschwächt, um eine natürlichere Wirkung zu erhalten.

se Probleme lassen sich durch Bildverarbeitung und Domänenwissen bzw. durch die Wahl einer geeigneten *Axial Plane*[7] im 3D-Datensatz mindern, aber nicht sicher lösen.

Nachdem der Arzt eine Gefäßkontur mit Gefäßkrankheiten skizziert hat, muss diese auf das Gitter der Blutfluss-Simulation übertragen werden. Hierzu werden alle Gitterzellen, die von der Gefäßkontur berührt werden, in der Simulation als *Hindernis-Zelle* betrachtet. Dies kann auch während des Zeichnens geschehen, wodurch der Einfluss des Skizzierens auf die Simulation unmittelbar zu sehen ist.

3.2.3 *Blutfluss manipulieren*

Der Blutfluss wird, wie in Abschnitt 3.1.2.1 beschrieben, durch das Einwirken einer externen Kraft manipuliert. Diese externe Kraft soll der Nutzer möglichst flexibel bestimmen können. Dazu wird ihm ein *Kraft*-Werkzeug zur Verfügung gestellt, welches wie die anderen Funktionen skizzenbasiert verwendet werden kann. Der Nutzer kann eine beliebige Linie zeichnen, welche durch einen Pfeil repräsentiert wird, der in die Zeichenrichtung zeigt. Dies gibt dem Arzt und Patient klare Informationen darüber, an welcher Stelle die Kraft auf welche Weise wirkt.

Für die Manipulation der Simulation muss die eingezeichnete Kraft auf das Geschwindigkeitsvektorfeld übertragen werden. Naheliegend wäre nun die Geschwindigkeitsvektoren zu beeinflussen, welche direkt unter der Zeichnung der Kraft liegen. Wird die externe Kraft so auf das Geschwindigkeitsvektorfeld ausgeübt, ergibt sich keine adäquate Abbildung dieser, da der Blutfluss nur in einer feinen Linie manipuliert wird. Stattdessen wird um den eingezeichneten Pfeil eine Region festgelegt, in welcher die Kraft wirken soll. Die Größe der Region ist dabei variabel. Es bietet sich an, hierfür die konstant festgelegte Gefäßbreite zu verwenden, da somit eine Krafteinwirkung für ein Gefäß durch das Skizzieren *eines* Pfeils erreicht werden kann. Durch die Abschwächung der Kraft am Rand der Region kann eine natürlich wirkende Kraft dargestellt werden. Für die Abschwächung wird die Gauß-Funktion genutzt, wodurch die Kraft im Zentrum (also da, wo der Nutzer gezeichnet hat) am stärksten wirkt und zum Rand schwächer wird. Der Effekt ist in Abbildung 31 illustriert.

7 Die *Axial Plane* beschreibt eine frei im Raum ausgerichtete Ebene (siehe [65]).

3.2.4 *Blutfluss erstellen*

Der Nutzer kann Blut bzw. Tinte mithilfe eines *Tinten*-Werkzeugs in der Anwendung platzieren. Hierzu tippt der Nutzer auf eine gewünschte Stelle, an der ein farbiger Kreis platziert wird. Unter der Annahme, dass der Arzt beim erneuten Platzieren eines Kreises dem Patienten die Mischung und Verwirbelung von Blutfluss zeigen möchte, wird dabei eine neue Farbe gewählt. Die Farbauswahl des Kreises ermittelt sich hierbei aus dem *Lab*-Farbraum durch das in Abschnitt 3.1.3 beschriebene Verfahren. Der Durchmesser des Kreises wird auf die Gefäßbreite festgelegt. Somit ist es möglich mit einem platzierten Kreis die Flussfarbe über die gesamte Gefäßbreite festzulegen. Der farbige Kreis wird zur Anwendung in der Blutfluss-Simulation auf das in Abschnitt 3.1.3 beschriebene Skalarfeld übertragen. Die skalaren Werte werden in der Region, in der der farbige Kreis gezeichnet wurde, erhöht. Durch Festlegung der Farbe für das Skalarfeld kann es nun visualisiert und durch Diffusion und Advektion im Gitter verteilt werden.

3.2.5 *Behandlungsmethoden*

Für die Behandlung von Gefäßkrankheiten sollen *Coils*, *Clips* und *Stents* gezeichnet werden. Die Methoden dafür werden folgend genauer beschrieben.

COILING Zum Platzieren eines Coils skizziert der Nutzer mithilfe eines *Coiling*-Werkzeugs direkt den Verlauf des Drahtes. Durch die Wahl eines kleinen Distanzparameters (siehe Abschnitt 3.2.1) können genaue Verläufe gezeichnet werden. Weiterhin wird durch die bereits beschriebene Resampling-Methode gewährleistet, dass der Draht keine Zickzack-Form erhalten kann, sondern sich beim Zeichnen abrundet. Neben dieser einfachen, aber direkten Methode einen Coil zu platzieren, sind weitere Möglichkeiten denkbar, die dem Nutzer Arbeit abnehmen. So könnte der Nutzer einen Bereich umranden, woraufhin das System selbstständig den Draht verteilt. Eine weitere Methode wäre denkbar, in welcher der Nutzer den Stift an einer Stelle absetzt, woraufhin von diesem Punkt aus ein Draht *sprießt* und sich in der näheren Umgebung verteilt. Diese Methoden bieten zum einen den Vorteil, dass der Nutzer nicht darauf achten muss, nicht über den Rand zu zeichnen. Zum anderen kann dadurch ein Aneurysma schneller mit Coils gefüllt werden. Dennoch wird die erste, direkte Methode genutzt, da diese einfach vom Nutzer anzuwenden ist.

CLIPPING Um einen Clip abzubilden, muss eine Klemme repräsentiert werden, welche verschiedene Längen haben kann. Dem Nutzer wird dies ermöglicht, indem das Verfahren zum Zeichen einer Linie von *Sketchpad* [89] übernommen wird. Der Punkt, bei dem der Nutzer das Skizzieren beginnt, repräsentiert den ersten Punkt des Clips. Nun führt der Nutzer den Stift weiter, wobei der Endpunkt des Clips unter dem Stift festgelegt wird. Sobald der Nutzer den Stift absetzt, wird der Clip mit den so ermittelten Start- und Endpunkten erstellt. Dieses Verfahren ermöglicht dem Nutzer, während der Platzierung die Ausrichtung des Clips anzupassen. Daher wirkt sich der Clip erst *nach* dem Zeichnen auf die Blutfluss-Simulation aus. Dies wird dem Nutzer illustriert, indem der Clip bis zur Platzierung durch eine gestrichelte Linie dargestellt wird.

STENTING Ein Anwendungsfall für einen Stent ist die Platzierung in einem Gefäß, welches ein Aneurysma enthält. Das Verfahren zum Skizzieren, mit welchem der Arzt den Stent in einem Gefäß platziert, soll sich an den realen Eingriff anlehnen: Dort wird über einen Ballonkatheter der Stent im Gefäß ausgedehnt. Der Nutzer zeichnet also in der Mitte des Gefäßes eine Linie, woraufhin die Anwendung die Berechnung der Stentausdehnung übernimmt. Die Modellierung dieses Verhaltens stellt eine Herausforderung dar, da der Stent sich entsprechend des zur Verfügung stehenden Platzes ausbreiten soll, dabei aber *nicht* in ein Aneurysma eindringen soll. Da die Anwendung die Gefäßkontur nicht semantisch unterscheidet, muss das gewünschte Verhalten anders erhalten werden. Die Grundidee besteht darin, ausgehend von der Zeichnung des Nutzers die umliegende Region zu analysieren und daraus Werte für die Breite des Stents abzuleiten. Das Verfahren wird ausführlich in Abbildung 32 beschrieben.

Die für das Verfahren nötigen Normalen können durch die Betrachtung der Nachbarpunkte erhalten werden. Gerade am Anfang und Ende der Skizze kann die Normale durch ungenaues Zeichnen nicht orthogonal zur Gefäßwand stehen. Um eine möglichst rechtwinklig-stehende Normale zu erhalten, wird eine größere Nachbarschaft bei der Normalenberechnung betrachtet.

Der Vorteil dieses Verfahrens ist, dass es *fehlertolerant* auf verschiedene Ausgangslinien des Nutzers reagiert. Zwar werden die besten visuellen Ergebnisse erzielt, wenn der Nutzer die Linie möglichst mittig in das Gefäß einzeichnet, dennoch führt ein ungenaues Einzeichnen nur zum leichten Eindringen des Stents in das Aneurysma. Nachteilig an dem Verfahren ist, dass es von dem Start- und Endpunkt der skizzierten Linie abhängt. Treffen die Normalen dieser Punkte auf keine Gefäßkontur (z. B. weil der Nutzer außerhalb eines Gefäßes anfängt zu zeichnen), so kann die Platzierung des Stents nicht berechnet werden.

Komplexere Möglichkeiten, um eine automatische Anpassung des Stents zu erhalten, sind die Nutzung eines *Feder-Masse-Modells* oder die Nutzung sog. *Snakes* [45]. Mit einem Feder-Masse-Modell kann eine Entfaltung des Stents unter bestimmten Bedingungen beschrieben werden. Die Bedingungen würden das Stoppen an einer Gefäßwand und eine maximale Entfaltungsbreite umfassen. Snakes (auch *active contours* genannt) basieren auf *Energie-minimierenden Splines*, welche genutzt werden könnten, um sich an die Gefäßwand anzuschmiegen.

Eine zusätzliche nützliche, aber nicht umgesetzte Behandlungsmethode ist die Verwendung von *Stents bei Stenosen*. Eine Möglichkeit für die Umsetzung sei dennoch konzeptionell beschrieben. Grundsätzlich lehnt sich das Verfahren an das für die Stentplatzierung im Aneurysma an. Der Nutzer zeichnet zunächst eine Linie durch die Stenose. Nun werden an dieser Linie in beiden Seiten Normalen gebildet, die die Länge der konstant definierten Gefäßbreite haben. Nun wird jede Normale auf einen Schnitt mit der Gefäßkontur geprüft. Tritt dieser auf wird nun, im Gegensatz zum Verfahren beim Aneurysma, nicht die Normale verändert, sondern die Gefäßkontur. Diese kann auf das Ende der Normale *geschoben* werden. Angewendet auf alle Normalen führt dieses Verfahren zur Ausdehnung der Stenose.

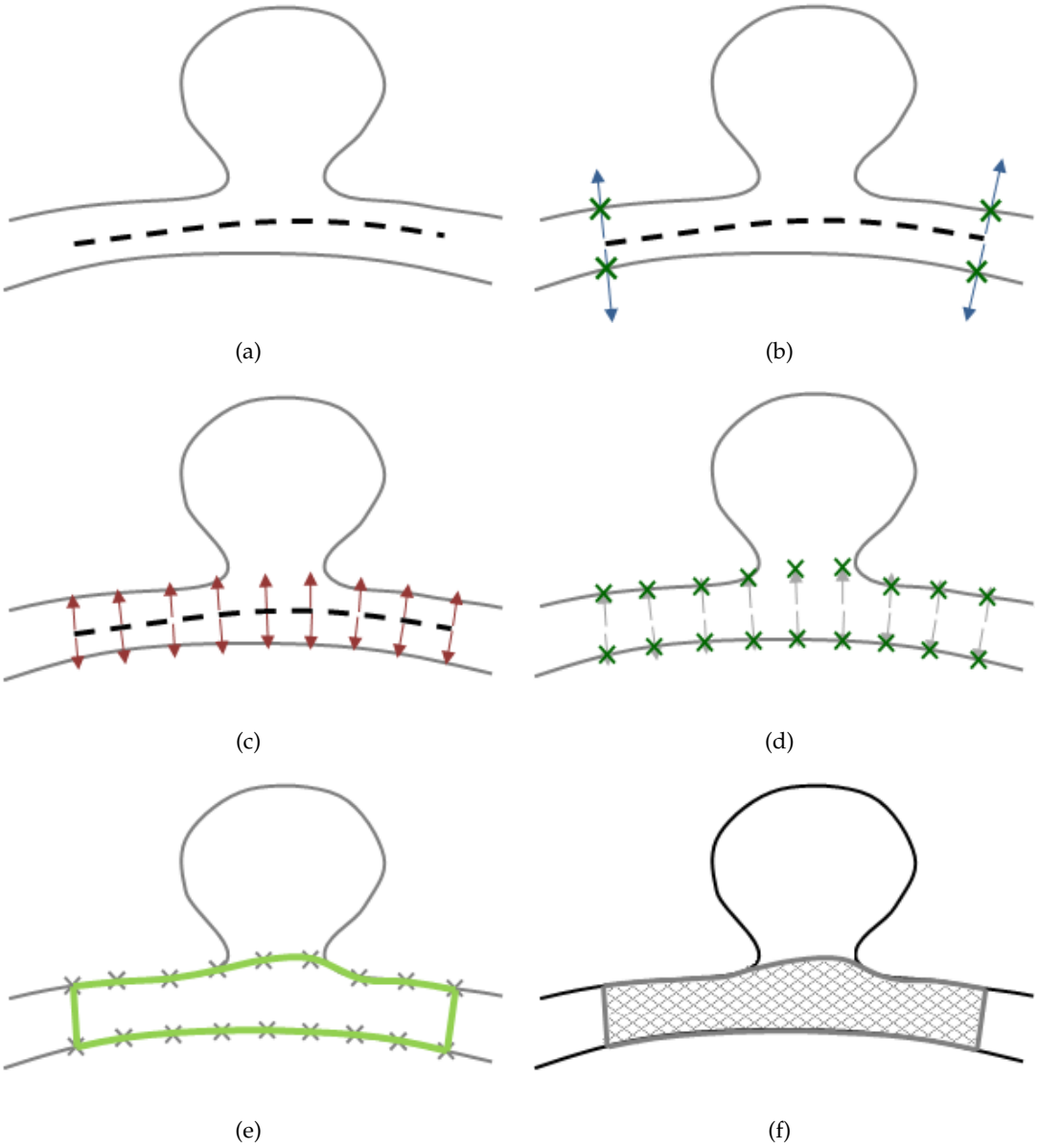


Abbildung 32: In (a) ist die skizzierte Linie des Nutzers in einem Gefäß mit Aneurysma dargestellt. Nachdem der Nutzer den Stift anhebt und so signalisiert, dass der Stent platziert werden soll, werden am Start- und Endpunkt der Skizze die Normalen der Linie bestimmt (b). Die Normalen haben dabei die Länge der konstanten Gefäßbreite. Für alle vier Normalen wird überprüft, ob sie die Gefäßkontur schneiden. Trifft dies nicht zu, wird der Stent verworfen. Konnte für alle Normalen ein Schnittpunkt bestimmt werden, wird die Länge des am weitesten entfernten Schnittpunktes l ermittelt. Nun werden für jeden Punkt der eingezeichneten Linie die Normalen bestimmt (c). Für jede Normale wird überprüft, ob sich diese mit der Gefäßkontur schneidet. Tritt dieser Fall ein, wird die Normale auf diesen Schnittpunkt beschränkt, andernfalls nimmt sie die Länge l an (d). Die durch dieses Verfahren erhaltenen Schnittpunkte werden nun verbunden (e) und bilden den Rahmen für den resultierenden Stent, welcher in (f) dargestellt ist.

3.2.6 *Objekte editieren und löschen*

Die Möglichkeit erstellte Objekten wie Clips, Coils oder Stents editieren oder löschen zu können adressiert primär das Usability-Kriterium der *Steuerbarkeit*. Das Editieren von Objekten wird im Gegensatz zum Löschen nur konzeptionell beschrieben.

Für das Löschen und Editieren muss der Nutzer eine Möglichkeit erhalten, die erstellten Objekte auszuwählen. Diese Möglichkeit muss sich dabei vom normalen Zeichnen unterscheiden, um die Nutzerabsicht klar abbilden zu können. Tippt der Nutzer bspw. auf einen Coil kann er diesen editieren wollen, oder beabsichtigen, einen Weiteren zu zeichnen. Durch die in Abschnitt 3.2.1 genannten Einschränkungen kann für diese Unterscheidung keine rechte Maustaste, kein Knopf an einem Stift oder die Verwendung einer multiplen Eingabe genutzt werden. Zur Unterscheidung der Nutzerintention zwischen *Erstellen* und *Bearbeiten* sind mehrere Methoden möglich:

1. Ein *Doppelklick*, der mit dem Stift ausgeführt wird,
2. Drücken und Halten des Stifts oder
3. die Verwendung von Gesten.

Die dritte Variante setzt die Verwendung eines *Recognizers* voraus, weswegen diese nicht genutzt wird. Stattdessen wird die zweite Variante verwendet.

Hält der Nutzer nun den Stift über ein Objekt, könnte sich nun ein Kontextmenü in Form von *inplace Menüs* oder *Shadow Buttons* öffnen (siehe Grundlagenabschnitt 2.3.5.1). Nun könnten abhängig vom ausgewählten Objekt verschiedene Parameter beeinflusst werden. So könnte die Dicke eines Clips oder die Blutdurchlässigkeit von Coils und Stents angepasst werden. Für platzierte Tinte könnte die Farbe geändert werden und für die eingezeichnete Kraft könnte die Breite des Bereichs, in der sie wirkt, oder deren Stärke eingestellt werden.

Statt eines Kontextmenüs, welches durch Halten und Drücken auf ein Objekt erscheint, werden die Objekte über diese Funktion gelöscht. Hierdurch kann der Nutzer falsch gezeichnete Elemente entfernen oder ggf. neu platzieren. Würde der Gesten-basierte Ansatz weiter verfolgt werden, könnte wie in [3] eine *Scratch-out*-Geste zum Löschen von Objekten verwendet werden. Hierbei führt der Nutzer eine Zickzack-Bewegung über dem Objekt aus, die vom System erkannt wird.

3.2.7 *Laden und Speichern*

Eine weitere Funktion, die nur konzeptionell beschrieben wird, ist das Laden und Speichern von Skizzen. Hierdurch könnten *häufig nötige Strukturen* fest in der Anwendung hinterlegt und bei Bedarf aufgerufen werden. Weiterhin könnte diese Funktion *behandlungsbegleitend* genutzt werden, indem einem Patienten bei jedem Gespräch die gleichen (ggf. eigenen) Gefäßstrukturen gezeigt werden. Weiterhin könnte es die Kollaboration von Ärzten unterstützen, die erstellte Skizzen miteinander teilen könnten.

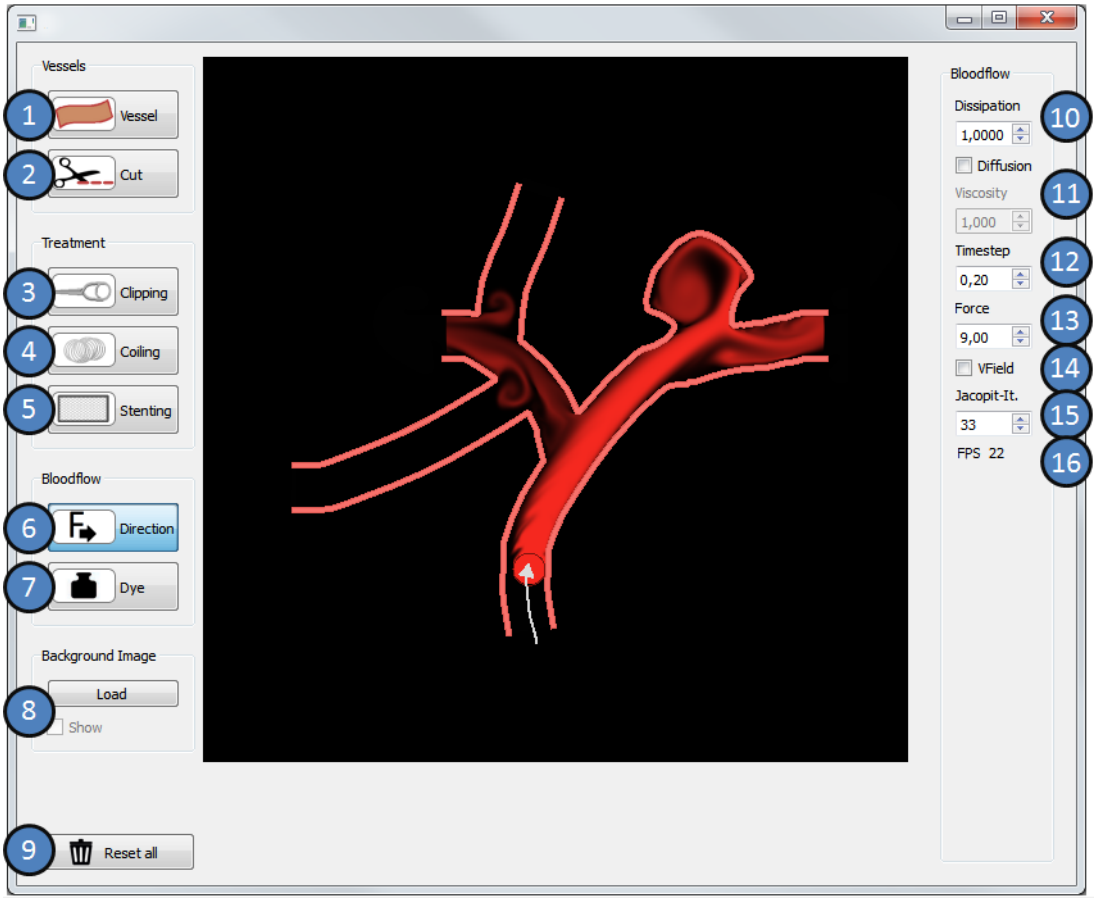


Abbildung 33: In der Abbildung ist das UI sowie eine gezeichnete Gefäßstruktur mit Blutfluss dargestellt. Auf der linken Seite befinden sich Werkzeuge zum: Erstellen (1) und Editieren (2) von Gefäßen, Erstellen von Clips (3), Coils (4) und Stents (5) sowie zum Manipulieren (6) und Visualisieren (7) des Blutflusses. Weiterhin kann ein Hintergrundbild (8) geladen werden. Mit dem *Alles zurücksetzen*-Werkzeug (9) lassen sich alle erstellten Objekte löschen. Auf der rechten Seite befinden sich Eingabemasken zur Veränderung der Parameter der Blutfluss-Simulation. Durch die „Dissipation“ (10) lässt sich eine Abschwächung der Blutflussstärke über eine Zeitspanne steuern. Mithilfe der Diffusionseingaben (11) lässt sich die Diffusion aktivieren/deaktivieren sowie die Viskosität einstellen. Mit dem Zeitschritt (12) lässt sich die Simulationsgeschwindigkeit Δt steuern (siehe Abschnitt 3.1.2.2). Mithilfe der Kraft (13) lässt sich die Stärke der vom Nutzer eingezeichneten Kraft variieren. Durch die „VField“-Checkbox (14) lässt sich der Fächerplot ein- bzw. ausblenden. Durch das Eingabefeld „Jacobi-It.“ lässt sich die Anzahl der Jacobi-Iterationen anpassen. In (16) wird dem Nutzer die Framerate angezeigt.

3.3 USER INTERFACE

Der Aufbau des UIs ergibt sich aus den beiden Überlegungen:

- Ein quadratisches Gitter für Blutfluss-Simulation zu nutzen und
- zur Eingabe ein direktes Zeigegerät zu verwenden.

Aus dem ersten Punkt resultiert, dass der Zeichenbereich, der identisch zum Gitter für die Simulation ist, quadratisch ist. Aufgrund des häufigen breitformatigen Formfaktors von Displays und Touch-Eingabegeräten ergibt sich freier Platz an den Seiten der Zeichenfläche. Dieser wird für die Platzierung der Menüs- und Eingabefelder verwendet. Die links- oder rechtsseitige Platzierung wird hierbei für eine semantische Unterscheidung der Funktionalität genutzt: Alle Funktionen

- zum Erstellen und Manipulieren von Gefäßen,
- zum Skizzieren von Behandlungen und
- zur Visualisierung und Steuerung des Blutflusses

werden auf der linken Seite platziert. Alle Parameter, mit denen die Simulationseigenschaften *verändert* werden können, befinden sich auf der rechten Seite.

Aus der zweiten Überlegung ergeben sich Anforderungen an die Bedienung. Um eine einfache Steuerung der Anwendung zu ermöglichen, werden alle Funktionen direkt anwählbar gemacht und es wird auf Untermenüs verzichtet. Um einen schnellen und fehlerfreien Wechsel von Zeichenmodi zu gewährleisten, werden die Knöpfe zum Umschalten möglichst groß dargestellt. Zusätzlich zur Beschriftung erhalten sie ein Symbol, welches die Identifikation der Funktion erleichtert. In Abbildung 33 wird das UI und dessen Funktionalität genauer beschrieben.

4

IMPLEMENTIERUNG

Dieses Kapitel gibt einen Überblick über die Implementierung der Anwendung. Dabei wird auf eine vollständige Erläuterung verzichtet. Stattdessen werden die grundlegenden Komponenten der Anwendung beschrieben und dabei exemplarisch umgesetzte Funktionen erläutert.

4.1 FRAMEWORK

Für die Implementierung wird das *Qt*-Framework genutzt, welches die plattformübergreifende Entwicklung von GUIs in *C++* unterstützt.[1] Folgende Funktionalitäten erleichtern dabei die Entwicklung des im Konzeptkapitels beschriebenen Prototyps:

- der *Qt Designer*[2], ein Werkzeug zur Erstellung des UIs,
- das *Graphics View Framework*[3], welches als Grundlage für die Zeichenfunktionalität verwendet wird und
- das *Qt OpenGL Modul*[4], welches die Kommunikation mit der GPU vereinfacht.

Der *Qt Designer* ermöglicht durch das *WYSIWYG*-Prinzip[5] die Erstellung der Benutzungsoberfläche. Mit ihm ist es möglich bildhaft Menüs und weitere Eingabeelemente zu erstellen, deren Funktionalität vorerst mit sog. *Stubs*[6] gefüllt wird.

Das *Graphics View Framework* bietet die grundlegende Logik, um die Interaktion und Verwaltung von einer großen Anzahl von individuellen grafischen Objekten zu ermöglichen. Die individuelle Gestaltung dieser Objekte erlaubt eine flexible, visuelle Repräsentation. Somit eignet sich das *Graphics View Framework* als Grundlage für die Skizzierungsoberfläche.

Die Funktionen der GPU werden mithilfe der *OpenGL Shading Language*[7] (kurz *GLSL*) angesprochen. Diese ermöglicht das Ausführen eigener Programme mittels *OpenGL* durch sog. *Shader*. Für die Kommunikation zwischen der CPU und der GPU

1 http://qt-project.org/, zuletzt aufgerufen: 08.03.2014.
2 http://qt-project.org/doc/qt-5.0/qtdesigner/qtdesigner-manual.html, zuletzt aufgerufen: 08.03.2014.
3 http://qt-project.org/doc/qt-5.0/qtwidgets/graphicsview.html, zuletzt aufgerufen: 08.03.2014.
4 http://qt-project.org/doc/qt-5.0/qtopengl/qtopengl-index.html, zuletzt aufgerufen: 08.03.2014.
5 Das *W*hat *Y*ou *S*ee *I*s *W*hat *Y*ou *G*et-Prinzip beschreibt ein System, in welchem das Erscheinungsbild des Inhalts beim Editieren stark mit dem korrespondierenden Ergebnis zusammenhängt.
6 *Stubs* bzw. *Stummel* bezeichnen vorerst verwendeten Programmcode, der als Stellvertreter für später zu implementierenden Programmcode steht.
7 https://www.opengl.org/documentation/glsl/, zuletzt aufgerufen: 08.03.2014.

müssen bestimmte Konstrukte (z.B. ein Renderkontext, ein Renderfenster usw.) initialisiert werden. Hierfür bietet das *Qt OpenGL Modul* geeignete Funktionen.

Die Implementierung des Prototyps gliedert sich in drei Teile:

- Die Realisierung der Blutfluss-Simulation auf der GPU,
- die Erstellung einer Skizzierungsoberfläche und Verarbeitung der Nutzereingaben auf der CPU und
- den echtzeitbasierten Austausch des Skizzierten mit der Simulation.

Im Folgenden werden diese einzelnen Komponenten in dieser Reihenfolge genauer beschrieben.

4.2 BLUTFLUSS-SIMULATION

Um die Simulation von Flüssigkeit auf der GPU zu realisieren, muss zunächst die Arbeitsweise der Grafikkarte untersucht werden. Zur Beschreibung eignet sich die *Renderpipeline*. Zwei wesentliche aufeinanderfolgende Schritte dieser Pipeline sind die *Verarbeitung von Geometrie* und das *Rendering durch Rasterung*. Beide Schritte könnten dabei für die Blutfluss-Simulation verwendet werden. Der Geometrie-verarbeitende Schritt nimmt Polygone und deren Vertices auf. Um diesen Schritt für die Simulation zu nutzen, müssten die verwendeten Gitter (Geschwindigkeit, Druck usw.) mithilfe von einem *kartesichen Gitter* aus Vertices beschrieben werden. Zusätzlich müssten für jeden Vertex Eigenschaften gespeichert werden (z. B. x- und y-Richtung der Geschwindigkeit oder die RGB-Komponenten einer Farbe). Alternativ kann der Rendering-Schritt genutzt werden, welcher für die Bildausgabe zuständig ist. Die nötigen Gitter werden dabei durch einzelne Bilder repräsentiert und die Gitterzellen entsprechen den Pixeln. Die Eigenschaften der Gitterzellen können über die Farbinformationen der Pixel gespeichert werden. Da sich der zweite Ansatz natürlicher in die Grundfunktionalität der Grafikkarte einfügt, wird dieser verwendet.

Als Beispiel für die Nutzung der GPU soll die Verarbeitung des *Geschwindigkeitsvektorfeldes* dienen. Zunächst müssen die Eigenschaften dieses Vektorfeldes so beschrieben werden, dass sie durch ein *Shader*-Programm von der GPU verarbeitet werden können. Hierfür eignen sich zweidimensionale Texturen. Diese stellen in OpenGL Objekte dar, die eine bestimmte *Größe* und ein bestimmtes *Format* besitzen. Als Parameter für die *Größe* dient hierbei die Anzahl der Vektoren im Geschwindigkeitsvektorfeld. Somit wird jeder Vektor durch einen Pixel in der Textur repräsentiert. Das *Format* der Textur wird nun entsprechend der nötigen Informationen für den Vektor gewählt. Jeder Vektor besitzt zwei Komponenten (eine x- sowie y-Geschwindigkeit), weswegen für die Textur zwei Farbkanäle bzw. ein Rot-Grün-Format gewählt wird.

Für die einzelnen Phasen der Blutfluss-Simulation (Advektion, Diffusion usw., siehe Konzeptabschnitt 3.1.2, Abb. 19) werden nun jeweils einzelne Shader-Programme verwendet. Bei der Verwendung von Texturen muss beachtet werden, dass diese entweder in einem Shader-Programm ausgelesen oder als Ausgabeziel (engl. *Rendertarget*) genutzt werden. Das gleichzeitige *Schreiben* und *Lesen* einer Textur ist also nicht möglich. Soll nun das Geschwindigkeitsvektorfeld z. B. advektiert werden, so wä-

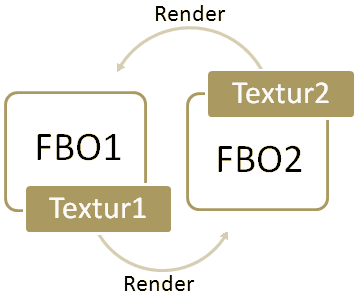


Abbildung 34: Schematisch dargestellt ist das alternierende Ping-Pong-Rendering zwischen zwei FBOs und deren Texturen.

re dieser gleichzeitige Zugriff nötig.[8] Um dennoch Texturen verwenden zu können besteht eine Möglichkeit darin, ein anderes Rendertarget zu nutzen. Hierzu kann das Display selbst oder ein sog. Framebuffer Object (FBO) genutzt werden. Während die Ausgabe auf dem Display die eigentliche Ausgabe stören könnte, wird die für den Nutzer unsichtbare Variante mit FBOs genutzt. Der Ablauf für einen Advektionsschritt des Geschwindigkeitsvektorfeldes ist also folgender:

1. Geschwindigkeitstextur an ein Shader-Programm binden,
2. als Rendertarget ein FBO auswählen,
3. das *Advektions*-Shader-Programm ausführen.

Das Ziel ist nun, das Renderergebnis, welches im FBO enthalten ist, zurück in die Geschwindigkeitstextur zu übertragen. Naheliegend wäre, das Ergebnis zu kopieren.[9] Die Berechnungsdauer hängt dabei von der Texturgröße ab. Da der Kopiervorgang auf der CPU durchgeführt wird, kann dieser bei großen Texturen einen *Flaschenhals* darstellen. Eine Alternative, die konstante Berechnungszeiten unabhängig von der Texturgröße besitzt, ist die Verwendung des sog. *Ping-Pong-Renderings*. Dabei werden jeweils zwei Texturen und zwei FBOs für das Geschwindigkeitsvektorfeld verwendet; die ausgelesene Textur und das verwendete FBO wechseln sich nach jedem Renderschritt ab (siehe Abb. 34). Nachteilig an diesem Verfahren ist, dass durch die zusätzlichen Texturen und FBOs ein größerer Speicheraufwand nötig ist sowie sich durch die Organisation der Objekte die Komplexität der Anwendung erhöht. Der Speicheraufwand ist minimal und kann daher vernachlässigt werden. Die Erhöhung der Komplexität der Anwendung ist durch den erhaltenen Berechnungsgeschwindikgeits-Vorteil gerechtfertigt.

Das beschriebene Verfahren wird grundlegend für alle Texturen und Berechnungsschritte verwendet. So werden jeweils zwei Texturen und FBOs erstellt für:

- das Geschwindigkeitsvektorfeld,
- die Skalarfelder der Farbe und

8 Zum einen wird mit dem Geschwindigkeitsvektor die Advektionsposition bestimmt (*Lesen*) und der alte Geschwindigkeitsvektor überschrieben (*Schreiben*).

9 Dies ist bspw. durch die OpenGL-Funktion *glCopyTexSubImage2D*, welche auf der CPU ausgeführt wird, möglich.

- das Skalarfeld des Drucks.

Weitere Texturen werden für die *Divergenz*, die *Hindernisse* und die eigentliche *Ausgabe* benötigt. Da für diese Texturen jedoch nicht das gleichzeitige Auslesen und Schreiben nötig ist, genügt es jeweils eine Textur und ein FBO zu erstellen.

In den jeweiligen Shader-Programmen können nun die Berechnungen der Navier-Stokes-Gleichungen unter Zuhilfenahme der FDM berechnet werden. Stellvertretend sei das Shader-Programm näher beschrieben, welches eine Jacobi-Iteration durchführt und somit neben der Diffusion auch die Berechnung des Drucks ermöglicht. Die im Konzeptabschnitt 3.1.2.4 beschriebene Formel sei zur Übersicht erneut erwähnt:

$$p_{i,j}^{(k+1)} = \frac{p_{i-1,j}^{k} + p_{i+1,j}^{k} + p_{i,j-1}^{k} + p_{i,j+1}^{k} + d_{i,j}}{4}.$$

$p_{i,j}^{(k+1)}$ beschreibt nun die Ausgabe eines Pixels des Shader-Programms, welcher auf das gebundene FBO gerendert wird. p^k und d sind die Texturen des *Drucks* und der *Divergenz*. Die Indizes beschreiben nun die x- und y-Koordinate eines Pixels. Da ein Shader-Programm für jeden Pixel eines FBOs ausgeführt wird, ist der momentane Pixel durch i und j beschrieben. Die Nachbarn können relativ zu diesem mit der Funktion:

```
texelFetchOffset(texture, coordinate, levelOfDetail = 0, offset);
```

bestimmt werden. Das Shaderprogramm für eine Jacobi-Iteration hat dann folgende Form:

```
out vec4 output; //Ausgabe-Farbe
uniform sampler2D Pressure; //Druck-Textur
uniform sampler2D Divergence; //Divergenz-Textur

void main()
{
    //bestimme momentane Koordinate
    ivec2 currentCoord = ivec2(gl_FragCoord.xy);

    //bestimme Nachbarn
    vec4 pNorth = texelFetchOffset(Pressure, currentCoord, 0, ivec2( 0,  1));
    vec4 pSouth = texelFetchOffset(Pressure, currentCoord, 0, ivec2( 0, -1));
    vec4 pEast  = texelFetchOffset(Pressure, currentCoord, 0, ivec2( 1,  0));
    vec4 pWest  = texelFetchOffset(Pressure, currentCoord, 0, ivec2(-1,  0));

    vec4 divergence = texelFetch(Divergence, T);

    //berechne Ausgabe-Farbe
    output = (pWest + pEast + pSouth + pNorth + divergence) * 0.25;
}
```

Dieses Shaderprogramm wird nun in einer Schleife aufgerufen, wobei sich das Ergebnis für die Poisson-Gleichung mit jeder Iteration des Shader-Programms verbessert.

4.3 SKIZZIEREN

Als Grundlage für die Skizzierungsoberfläche dient eine Qt-Klasse aus dem *Graphics View Framework*: die *QGraphicsScene*. Diese Szene stellt einen Container für Anzeige

und Verwaltung von *QGraphicsItems* dar. Diese Objekte können grafische Primitive wie Linien oder Kreise darstellen, aber auch dazu verwendet werden, *individuelle Objekte* anzuzeigen. Daher eignet sie sich als Basis-Klasse für die *Repräsentation* von den gewünschten Objekten (z.B. Gefäße, Stents, Clips usw.). Um das Objekt für die Nutzung der GPU anzupassen, wird diese Klasse nicht direkt genutzt, sondern eine *SketchGPUItem*-Klasse erstellt, welche genauer im nächsten Abschnitt 4.4 beschrieben wird.

Neben der Repräsentation müssen die Objekte die im Konzeptabschnitt 3.2.1 beschriebene Logik zum *Resampling* und *Glätten* implementieren. Dafür wird das *LineObject*, eine zweite Basisklasse, erstellt. Das *LineObject* besitzt die Methode *addSketchPos(x, y)* mit der alle Koordinaten, die durch die Stifteingabe erzeugt werden, verarbeitet werden. Der erste eingehende Punkt wird dabei in einer Liste gespeichert. Jeder weitere Punkt wird verworfen, wenn dieser eine zu geringe Distanz d zum letzten Punkt in der Liste besitzt. Ist die Distanz d groß genug, so wird der Punkt in der Liste aufgenommen. Anschließend wird der *vorletzte* Punkt in der Liste mit einem Gauß-Filter mit i Iterationen geglättet. Durch die Kapselung der Funktionalität in diesem Objekt können neue Objekte von diesem Erben und die Parameter d und i objektspezifisch angepasst werden. So kann bspw. für ein Gefäß der Distanzparameter größer gewählt werden, da durch die Gefäßbreite weniger gesampelte Positionen nötig sind. Bei einem Coil hingegen ist ein kleinerer Abstand zwischen den gesampelten Punkten und eine höhere Iterationszahl des Gauß-Filters wünschenswert. Hierdurch kann der Nutzer einen detaillierten Coil zeichnen, der durch die häufigen Iterationen des Gauß-Filters eine gleichmäßige Krümmung besitzt. Aus der Liste der so erzeugten Punkte wird ein Pfad mithilfe der Qt-Klasse *QPainterPath* erstellt. Diese Klasse ermöglicht:

- eine Kontur von einem Pfad zu erhalten sowie
- andere Pfade vom momentanen Pfad zu *addieren* oder zu *subtrahieren*.

Wie diese Funktionen in der Anwendung verwendet werden, sei am Beispiel der Gefäßerstellung genauer beschrieben. Nachdem der Nutzer eine Linie skizziert hat und ein dazu erstelltes *LineObject* die Eingabepositionen neu gesampelt und geglättet hat, wird aus der erhaltenen Liste ein *QPainterPath* erstellt. Nun kann von diesem Pfad eine Kontur mit beliebiger Breite erstellt werden. Für die Breite wird die gewünschte Gefäßbreite verwendet. Die so erhaltene Kontur (welche wiederum durch einen *QPainterPath* beschrieben wird) besitzt nun die Form des gewünschten Gefäßes.

Zeichnet der Nutzer nun ein zweites Gefäß, so wird daraus erst (wie im vorherigen Absatz beschrieben) die Kontur generiert und diese zum ersten Gefäß *addiert*. Überschneiden sich die Gefäße, führt die Addition dabei zu der im Konzeptabschnitt 3.2.2 beschriebenen, kreuzungsfreien Struktur. Während durch das Hinzufügen neue Gefäßäste erstellt oder Aneurysmen gezeichnet werden können, kann das *Ausschneide*-Werkzeug mit der Subtraktions-Funktion des *QPainterPathes* realisiert werden. Hierzu wird der vom Nutzer mit dem *Ausschneide*-Werkzeug erzeugte Pfad von dem Pfad der Gefäßkontur abgezogen.

Im Konzeptabschnitt 3.2.6 wird die Interaktionsform vorgestellt, Objekte durch *Tippen-und-Halten* zu löschen. Da diese Interaktionsform von den meisten Betriebssystemen nicht nativ unterstützt wird, wird sie folgendermaßen implementiert. Setzt der Nutzer den Stift auf ein Objekt (dies erzeugt ein *mouseDown-Event*), wird ein

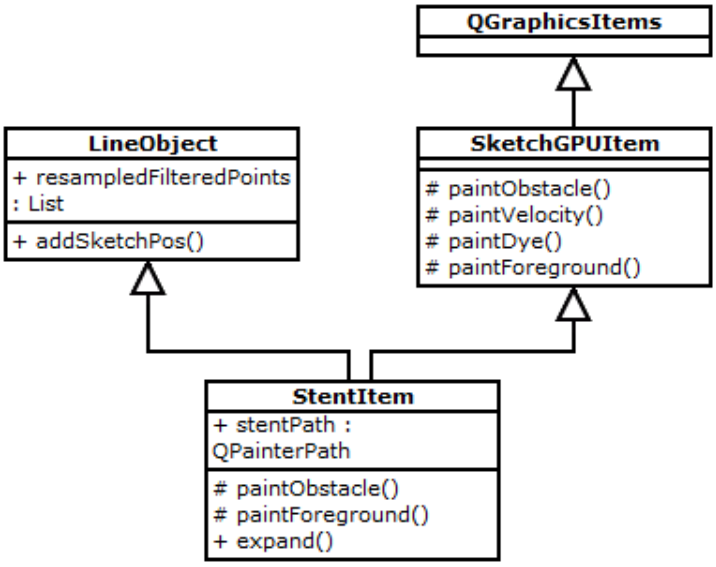


Abbildung 35: In der Abbildung ist ein *Klassendiagramm* dargestellt, um exemplarisch am Beispiel des *StentItems* die Vererbungshierarchie aufzuzeigen. Vom *LineObject* wird die Funktionalität zum Resampeln und Glätten der Eingabedaten geerbt und eine Liste mit den nicht verworfenen Daten erhalten. Aus dieser wird im *StentItem* durch die *expand()*-Funktion der entfaltete Stent erhalten. Mithilfe der Mehrfachvererbung werden über das *QGraphicsItem* durch das *SketchGPUItem* verschiedene Zeichenfunktionen geerbt. Von diesen sind für das *StentItem* nur zwei relevant: die Ausgabe-Repräsentation für den Nutzer (*paintForeground()*) und die Hindernis-Repräsentation (*paintObstacle()*).

Zeitgeber gestartet, der nach Ablauf das Objekt löscht. Sollte während des Zählens des Zeitgebers der Stift angehoben oder bewegt werden, wird der Zeitgeber gestoppt. Weiterhin muss das initiale *mouseDown*-Event nachgeholt werden. Dies wird erreicht, indem dies manuell erstellt und ausgelöst wird.

4.4 ZUSAMMENFÜHRUNG DER SKIZZIERUNGSOBERFLÄCHE UND DER SIMULATION

Die Informationen die durch die Skizzierungsoberfläche erhalten werden, können nicht direkt von der GPU verarbeitet werden. Hierzu ist eine Schnittstelle nötig, die folgend genauer beschrieben wird. Auf der Skizzerungsseite wird dem Nutzer ermöglicht u. a. Gefäße, Tinte und eine Krafteinwirkung auf das Vektorfeld zu zeichnen. Auf der Blutfluss-Simulations- und somit GPU-Seite müssen diese Elemente als Hindernis, platzierte Farbe und Geschwindigkeit interpretiert werden. Um dies zu erreichen, werden für jedes dargestellte Objekte verschiedene Repräsentationen geschaffen. Diese sind:

- Eine *Ausgabe*-Repräsentation - Diese ist für den Anwender sichtbar. Hierbei steht im Vordergrund, dem Nutzer die Funktion eines Objekts zu illustrieren.
- Eine *Hindernis*-Repräsentation - Diese wird von der GPU für die Einhaltung der *Randwertbedingungen* genutzt.
- Eine *Farb*-Repräsentation - Hiermit wird in den Texturen Tinte platziert.

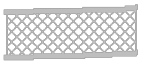
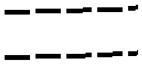
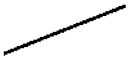
- Eine *Geschwindigkeits*-Repräsentation - Durch diese Darstellung wird das Geschwindigkeitsvektorfeld manipuliert.

Diese Darstellungen werden in der Klasse *SketchGPUItem* als leere Funktionen gekapselt. Objekte, die von dieser Klasse erben, können diese Funktionen überschreiben und so Einfluss auf die Blutfluss-Simulation nehmen. Dies sei am Stent-Objekt verdeutlicht, dessen Vererbungshierarchie in einem Klassendiagramm in Abbildung 35 dargestellt ist. Während der Nutzer den Stent skizziert, wird dieser nur in der Ausgabe-Repräsentation als gestrichelte Linie dargestellt und nimmt somit keinen Einfluss auf die Simulation. Nachdem der Stent erstellt wurde, wird er in der Ausgabe-Repräsentation durch seine *durchgezogene* Umrandung und ein innenliegendes Gitter beschrieben. Dies verdeutlicht dem Nutzer die Lage und das Aussehen des Stents. Für die Simulation ist die Gitterdarstellung hingegen unnötig und die durchgezogene Umrandung einschränkend, da kein Blut durch sie dringen kann. Daher wird in der Hindernis-Repräsentation nur die Umrandung mit einer *gestrichelten* Linie gezeichnet. In der Tabelle 1 ist eine Übersicht über alle Objekte und deren Repräsentationen enthalten.

Der verwendete Ansatz ist vorteilhaft, weil neue Objekte flexibel in das bestehende System eingepflegt und damit komplexe Funktionen modelliert werden können. So wäre bspw. eine Erweiterung mit einem Katheter-Objekt denkbar, welches Kontrastmittel in ein Gefäß einspritzt. In der Ausgabe-Repräsentation wird der Katheter für den Nutzer dargestellt. In der Farb-Repräsentation fügt der Katheter an seinem Ende ein andersfarbiges Kontrastmittel hinzu, welches gleichzeitig durch eine geeignete Geschwindigkeits-Repräsentation in eine Richtung bewegt wird.

Die Kommunikation von der Skizzierungsoberfläche mit der GPU funktioniert bei dem beschriebenen Prozess folgendermaßen: Während der Nutzer ein neues Objekt erstellt, werden alle Repräsentationsformen nacheinander gerendert. Das Ergebnis jedes Renderschrittes wird von der Skizzierungsoberfläche *übernommen* und in die zugehörige Textur der GPU übertragen. Nachdem alle Repräsentationen gerendert wurden, wird die Ausgabe-Repräsentation wiederhergestellt. Das Rendering der einzelnen Repräsentationen geschieht dabei so schnell, dass es für den Nutzer verborgen bleibt.

Tabelle 1: Die Tabelle zeigt eine Auflistung aller erstellbaren Objekte und ihr Aussehen in den jeweiligen *Renderstates*. Ein *x* kennzeichnet, dass ein Objekt in diesem Renderstate unsichtbar ist. Hervorzuheben sind einige Besonderheiten: Für die Gefäße wird der Ein- und Ausgang mit der Farbe Schwarz gezeichnet. Dies verhindert optisch das Aus- oder Eindringen von Flüssigkeit, da diese dabei schwarz gefärbt wird. Der Stent wird in der Hindernisdarstellung mit gestrichelten Linien gezeichnet, um leichten Blutfluss hindurch zu ermöglichen. Die Ausgabe der Tinte wird nicht vollständig ausgefüllt, da somit der Blick auf die eigentliche Färbung nicht verdeckt wird. Hierdurch kann z.B. das Pulsieren des Blutflusses gesehen werden.

	Ausgabe	Hindernis	Farbe	Geschwindigkeit
Gefäße				x
Stent			x	x
Clip			x	x
Coil			x	x
Tinte		x		x
Kraft		x	x	

5

EVALUIERUNG

In dieser Arbeit wird ein Konzept und die Implementierung eines Prototyps beschrieben, welcher Methoden zur Skizzierung von Gefäßen, deren Erkrankungen und Behandlungsmöglichkeiten bietet. Die Zielstellung dabei ist Methoden zu finden, die für die Anwendung in einem Arzt-Patienten-Dialog geeignet sind. Diese Eignung wird in Form der *Usability* beschrieben, welche u. a. durch die im Grundlagenkapitel 2.3.4 beschriebenen Usability-Tests untersucht werden kann.

Für die Evaluierung wird eine vereinfachte Version des Prototyps genutzt. Diese enthält nur Funktionen, die das Erstellen von Gefäßstrukturen, deren Behandlung und das Platzieren von Blut ermöglicht. Die Funktionen zum Löschen von erstellten Objekten und zur Einstellung der Parameter der Blutfluss-Simulation wurden ausgeblendet, um ein möglichst einfaches und übersichtliches UI zu bieten (siehe Abb. 36).

In der Evaluierung der vorliegenden Arbeit werden sowohl *qualitative* Methoden verwendet, um eine offenere Form der Diskussion mit den Probanden zu ermöglichen, als auch *quantitative* Methoden, um die Software-Ergonomie greifbarer zu untersuchen. Als Stellvertreter der qualitativen Methoden wird die *Think-Aloud-Methode* verwendet. Hierbei wird der Proband gebeten, während der Benutzung des Prototyps seine Gedanken verbal zu äußern. Dies ermöglicht Einblicke darin zu erhalten, wie der Nutzer das Programm wahrnimmt und an welchen Stellen Missverständnisse auftreten. Als quantitative Methode wird ein Fragebogen verwendet. Dieser ermöglicht durch gezielte Fragen Denkprozesse beim Probanden auszulösen. Beim Entwurf des Fragebogens wird sich an Bestehenden zur Untersuchung von Usability orientiert. Hierbei wurden folgende Fragebögen näher hinsichtlich Zugänglichkeit und Eignung für die zu testende Anwendung untersucht:

- Attrakdiff[1] [33],
- User Experience Questionaire[2],
- ISONORM 9241/110-Fragebogen,
- IsoMetrics[3],
- SUMI[4] (Software Usability Measurement Inventory) und
- QUIS[5] (Questionnaire for User Interface Satisfaction).

1 http://attrakdiff.de/, (zuletzt aufgerufen: 02.03.2014)
2 http://www.ueq-online.org/?lang=de, (zuletzt aufgerufen: 02.03.2014)
3 http://www.isometrics.uni-osnabrueck.de/index.htm, (zuletzt aufgerufen: 02.03.2014)
4 http://sumi.ucc.ie/academic.html, (zuletzt aufgerufen: 02.03.2014)
5 http://lap.umd.edu/quis/, (zuletzt aufgerufen: 02.03.2014)

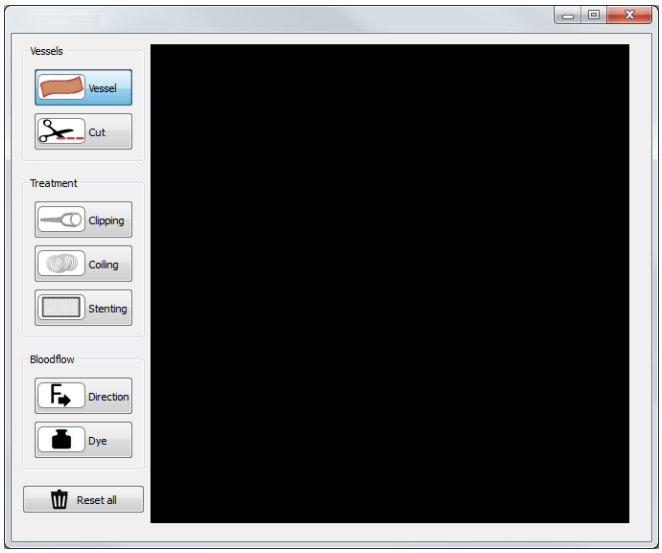


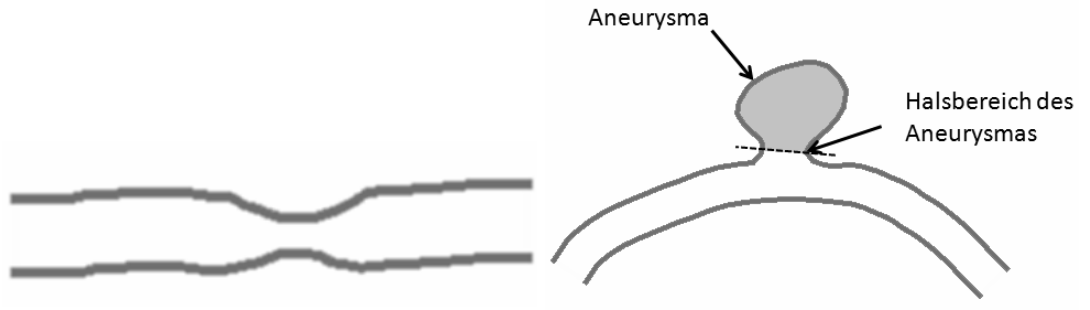
Abbildung 36: Das vereinfachte UI des Prototyps, welches für die Evaluierung verwendet wurde.

Nach Ausschluss der Varianten, die nicht direkt zugänglich sind[6] und nach zusätzlicher Rücksprache mit einem erfahrenen Forscher auf diesem Gebiet, wird der Fragebogen ISONORM 9241/110 verwendet. Von diesem Fragebogen existieren zwei Varianten, die durch die zusätzliche Bezeichnung *S* für kurz und *L* für lang gekennzeichnet sind. Als Grundlage wird die Langfassung ausgewählt. Der Fragebogen ISONORM 9241/110L gruppiert die Fragen nach allen Usability-Kriterien. Die einzelnen Fragen aus jeder Gruppe sind vom Probanden mithilfe einer ungeraden, siebenelementigen Likert-Skala auszufüllen.[7] Die ungerade Anzahl der Skala ermöglicht den Probanden auch neutrale Aussagen zu treffen.

Einige Fragen werden aus dem Original-Fragebogen entfernt, da diese bei der Befragung nicht relevant sind. Ein Beispiel aus dem Bereich Lernförderlichkeit ist die Frage nach der Ermutigung durch die Software „neue Funktionen auszuprobieren“. Da während der Evaluierung alle Funktionen vorgestellt und getestet werden, die in der Evaluierungsversion des Prototyps möglich sind, wäre es nicht möglich, diese Frage eindeutig zu beantworten. Weiterhin wird der Fragenblock *Individualisierbarkeit* vom Fragebogen entfernt. Dies ist der Tatsache geschuldet, dass der Prototyp keine Möglichkeit besitzt, die Anwendung individuell anzupassen. Diese Thematik wird im Abschlussabschnitt 6.2 der Arbeit erneut aufgegriffen. Die Bezeichnungen des Original-Fragebogens sw#, wobei # für die Fragenumnmer steht, wurden erhalten. Zusätzlich wurden demografische Fragen zum Alter, Geschlecht und Computernutzung mit Stiftinteraktion hinzugefügt. Der verwendete Fragebogen ist im Anhang Fragebogen zu finden.

6 Einige Fragebögen sind kostenpflichtig oder setzen eine Anmeldung voraus.

7 Die Likert-Skala wird verwendet, um die Einstellungen des Probanden zu einer Frage zu untersuchen. Hierzu wird ein Sachverhalt mit positiven oder negativen Aussagen gekennzeichnet, welchen der Proband zustimmen oder welche er ablehnen kann.



(a) (b)

Abbildung 37: In (a) ist die erste Aufgabe, das Zeichnen einer Stenose, abgebildet. In (b) soll der Proband ein Gefäß mit Aneurysma zeichnen und im Halsbereich clippen.

5.1 ABLAUF

Zu Beginn der Nutzerstudie wurde jedem Proband das Thema der Masterarbeit beschrieben sowie der Ablauf der Evaluierung erklärt. Dabei wurde jedem Teilnehmer mitgeteilt, dass Fragen während der gesamten Studie erlaubt sind. Zur Anwendung der *Think-Aloud*-Methode wurde um „lautes Denken“ während der Durchführung der Aufgaben gebeten. Darauf folgte die Vorstellung aller Funktionen des Programms, die für die Evaluierung relevant sind.

Alle Aufgaben wurden an einem *SmartBoard*[8] ausgeführt, welches durch Stiftinteraktion bedient wurde.

Nun wurde dem Teilnehmer ein Aufgabenblatt mit zwei *Übungsaufgaben* zum Erlernen der Grundfunktionen sowie drei *Aufgaben* überreicht (siehe Anhang Aufgabenblatt). Die Aufgaben werden folgend genauer beschrieben.

Bestandteil der ersten Aufgabe war es, das *Gefäß*- und *Ausschneide*-Werkzeug zur Erstellung einer Stenose zu nutzen (siehe Abb. 37(a)). In der zweiten Aufgabe sollten die Teilnehmer eine Gefäßstruktur mit einem Aneurysma erstellen und anschließend mithilfe des *Clipping*-Werkzeugs einen Clip im Halsbereich des Aneurysmas zeichnen (siehe Abb. 37(b)). Die letzte Aufgabe war mehrteilig. Zunächst sollte eine komplexere Gefäßstruktur mit einem Aneurysma gezeichnet werden (siehe Abb. 38(a)). Zusätzlich sollte das *Direction*- und das *Dye*-Werkzeugs genutzt werden, um den Blutfluss zu simulieren und visualisieren. Im nächsten Teil der Aufgabe sollte das Aneurysma unter Verwendung des *Coiling*-Tools gefüllt werden (siehe Abb. 38(b)). Im letzten Teil der dritten Aufgabe sollte ein Stent mit dem *Stent*-Werkzeug platziert werden (siehe Abb. 38(c)).

Nach Erfüllung aller Aufgaben wurden die Teilnehmer gebeten den beschriebenen Fragebogen auszufüllen (siehe Anhang Fragebogen).

5.2 AUSWERTUNG

In der Evaluierung wurden 14 Personen (vorwiegend Forscher mit medizinischem Hintergrund) befragt (3 weiblich, 11 männlich). Das mittlere Alter betrug 31 Jah-

8 Verwendet wurde das SMARTBoard E70; ein 70"-Bildschirm welcher Touch- und Stiftbedienung ermöglicht.

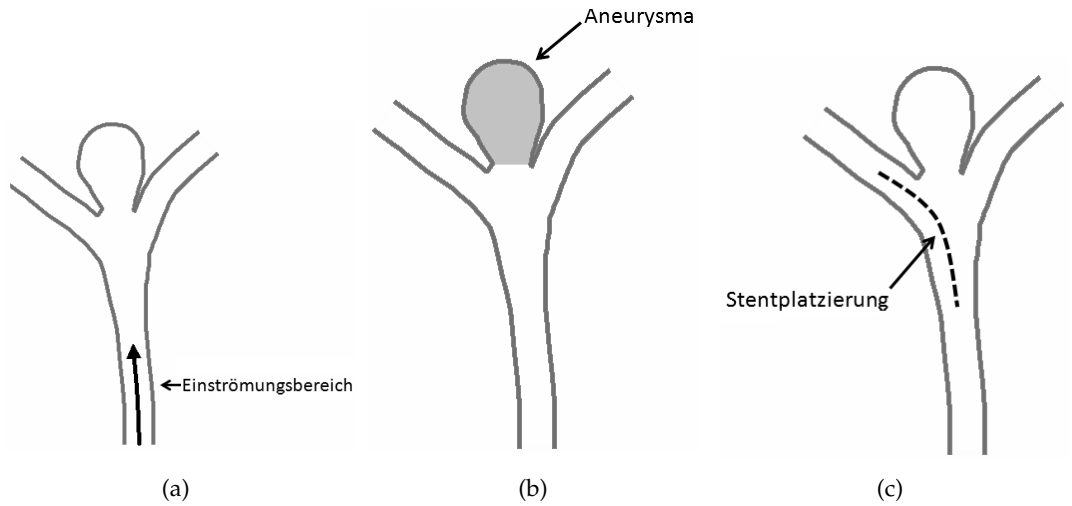


Abbildung 38: In (a) soll der Proband eine verzweigte Gefäßstruktur mit einem Aneurysma zeichnen. Zusätzlich soll er den Blutfluss im Einströmungsbereich simulieren und darstellen. (b) markiert den Bereich im Aneurysma, den der Nutzer mit einem Coil auszeichnen soll. Die letzte Aufgabe ist in (c) dargestellt: Hier soll der Proband einen Stent einzeichnen.

re (Minimum: 25, Maximum: 44). Die Probanden hatten durchschnittlich 20 Jahre Computer-Erfahrung (Minimum: 14, Maximum: 30) und 71% hatten bereits Erfahrung mit Stiftinteraktion. Die Dauer der Nutzerstudie für einen Probanden zur Erfüllung der Aufgaben und Ausfüllen des Fragebogens betrug zwischen 15-20 Minuten.

Zunächst folgt eine Auswertung der durchgeführten Think-Aloud-Methode. Die Aussagen der Probanden werden dabei nach den *einzelnen Werkzeugen* und *allgemeinen Aussagen* sortiert. Spezifische Aussagen von Probanden werden durch die Bezeichnung [P#] gekennzeichnet, wobei mit # die Probanden-ID kodiert ist. Darauf folgt die Auswertung der Fragebögen. Abschließend werden die Ergebnisse zusammengefasst und Möglichkeiten zur Verbesserung der Anwendung genannt.

5.2.1 *Auswertung der einzelnen Werkzeuge*

5.2.1.1 *Gefäß-Werkzeug*

Die grundlegende Funktion des Gefäß-Werkzeugs wurde von den Probanden positiv bewertet. Es ermöglichte „Gefäße und Aneurysmen nach der eigenen Vorstellung“ [P13] zu zeichnen. Als problematisch wurde beschrieben, dass durch die breite Darstellung eines Gefäßes das Zeichnen von Details erschwert werden kann. Ein weiterer Kritikpunkt betrifft die Bezeichnung des Werkzeugs: Während mit dem *Ausschneide*-Werkzeug eine Operation beschrieben wird, beschreibt das *Gefäß*-Werkzeug ein Objekt.

5.2.1.2 *Ausschneide-Werkzeug*

Bei der erstmaligen Verwendung des Schneide-Werkzeugs hatten fast alle Probanden Schwierigkeiten. Diese rührten aus verschiedenen Erwartungshaltungen zur Funktionsweise des Werkzeugs:

- Einige Probanden nutzten das Schneide-Werkzeug wie ein Rasiermesser und versuchten Gefäßäste durch die Skizzierung einer Linie *abzuschneiden*.
- Einige Teilnehmer versuchten das Werkzeug wie einen *Radierer* zu verwenden und so Gefäßstrukturen *wegzuzeichnen*.

Nach mehrmaliger Benutzung wurde die Funktionsweise jedoch besser verstanden und positiv bewertet. Ein Proband, der das Werkzeug wie einen Radierer verwenden wollte, merkte an, dass eine „viel genauere Eingaben als mit einem Radierer“ [P6] möglich sei. Ein Proband merkte an, das beim Ausschneiden von Bereichen *Reste* von Gefäßen übrig bleiben können, was die Blutfluss-Simulation beeinflussen kann.

5.2.1.3 *Clipping-Werkzeug*

Die Funktionsweise des Clipping-Werkzeugs wurde von allen Probanden sofort verstanden. Positiv hervorgehoben wurde die Möglichkeit, während des Zeichnens die Position des Endpunktes verändern zu können. Von Probanden die am falschen Punkt angefangen haben zu zeichnen wurde kritisiert, dass sie keine Möglichkeit haben die Position des Clips zu korrigieren.

5.2.1.4 *Coiling-Werkzeug*

Während alle Probanden das Coiling-Werkzeug sofort richtig nutzen konnten wurde kritisiert, dass das händische Ausfüllen eines Aneurysmas zu aufwendig ist. Weiterhin erlaubt die manuelle Platzierung auch das Zeichnen über Gefäßwände hinweg. Im Gegensatz dazu hatten einige Probanden Freude daran, den Coil manuell einzeichnen zu können, was für die *User Experience* der Anwendung spricht. Ein Proband hätte erwartet, dass das Blut zumindest teilweise den Coil durchdringt und nicht vollständig von ihm blockiert wird [P13].

5.2.1.5 *Stent-Werkzeug*

Der Stent konnte von den meisten Probanden ohne Schwierigkeiten genutzt werden. Positiv hervorgehoben wurde die Benutzung sowie das automatische Auffalten. Einige Probanden zeichneten aus dem Gefäß heraus, was dazu führte, dass kein Stent erstellt wurde. Hierbei wurde die fehlende Rückmeldung kritisiert. Eine weitere Anmerkung eines Probanden war, dass das Stent-Werkzeug fälschlicherweise ausgewählt und benutzt werden kann, selbst wenn kein Gefäß vorhanden ist [P7].

5.2.1.6 *Kraft-Werkzeug*

Während die Funktionsweise des Kraft-Werkzeugs verstanden wurde, konnten einige Probanden durch ungenaues Zeichnen die Kraft-Richtung nicht wie gewünscht zeichnen. Problemtisch war, dass kleine Fehler beim Zeichnen eine große Wirkung auf die Blutfluss-Simulation haben können. Weiterhin wurde angemerkt, dass die

Kopplung der Zeichenrichtung und Kraftrichtung dazu führen kann, dass sich ein Nutzer nicht immer für die einfacher zu zeichnende Richtung entscheiden kann. Zusätzlich wurde das verwendete Symbol des Knopfes kritisiert. Während die Darstellung eines Pfeils ausreichen würde, ist die zusätzliche Darstellung des Buchstaben F (für die *Kraft* bzw. *Force*) für Ärzte oder Patienten nicht sicher verständlich.

5.2.1.7 *Tinten-Werkzeug*

Die meisten Probanden versuchten das Tinten-Werkzeug, welches das Platzieren einer punktuellen Farbquelle erlaubt, über eine Strecke zu zeichnen. Nach dieser anfänglichen Überraschung wurde das Werkzeug aber verstanden und in Kombination mit dem Kraft-Werkzeug gerne genutzt. Kritisiert wurde, dass die Möglichkeit mehrere Farben zu verwenden von der Anwendung nicht kommuniziert wird.

5.2.2 *Allgemeine Aussagen*

Einige Probanden hatten Schwierigkeiten beim Zeichnen des Halses eines Aneurysmas. Der häufige Wechsel zwischen Gefäß- und Ausschneide-Werkzeug wurde dabei als störend empfunden. Positiv hervorgehoben wurde die übersichtliche Darstellung des UIs und die klare Anordnung der Werkzeuge. Hierbei wurde angemerkt, dass die Anordnung der Werkzeuggruppen nicht dem typischen Arbeitsfluss folgt. Der Arbeitsfluss sollte demnach sein:

- erst die Gefäße zu erstellen,
- dann den Blutfluss zu visualisieren und
- am Ende die Behandlungsmethoden einzuzeichnen.

Zusätzlich wurde von einem Probanden angemerkt, dass „mit dem Tool gut dargestellt werden kann, weshalb einige Eingriffe nicht möglich sind“ [P6]. Die generellen Aussagen zur Anwendung waren positiv und interessiert. Von der Mehrheit der Probanden wurde die Blutfluss-Simulation und deren Darstellung als optisch ansprechend und realitätsnah beschrieben.

5.2.3 *Auswertung der Fragebögen*

Für die Auswertung der Fragebögen werden zunächst die Mittelwerte der einzelnen Fragen gebildet. Somit ergibt sich für jede Frage die durchschnittliche Wertung über alle Probanden. Hierbei sei angemerkt, dass nicht alle Fragen von allen Probanden beantwortet wurden. In diesem Fall werden diese Probanden für die jeweilige Frage nicht für die Mittelwertberechnung betrachtet. Die Häufigkeit der getroffenen Aussagen ist für jede Frage in Abbildung 39 dargestellt,

Die Mittelwerte aller Fragen sind in Abbildung 41 dargestellt. Hierbei erhielten drei Fragen (sw08, sw26 und sw29) eine Bewertung zwischen 1 und 2 und sind somit die Negativsten. sw08 betrifft die Beschreibung von unzulässigen/zulässigen Eingaben. sw26 und sw29 sind aus dem Bereich *Fehlertoleranz* und beschreiben die Folgen von Eingabefehlern sowie deren Korrekturaufwand. Diese Ergebnisse liefern einen

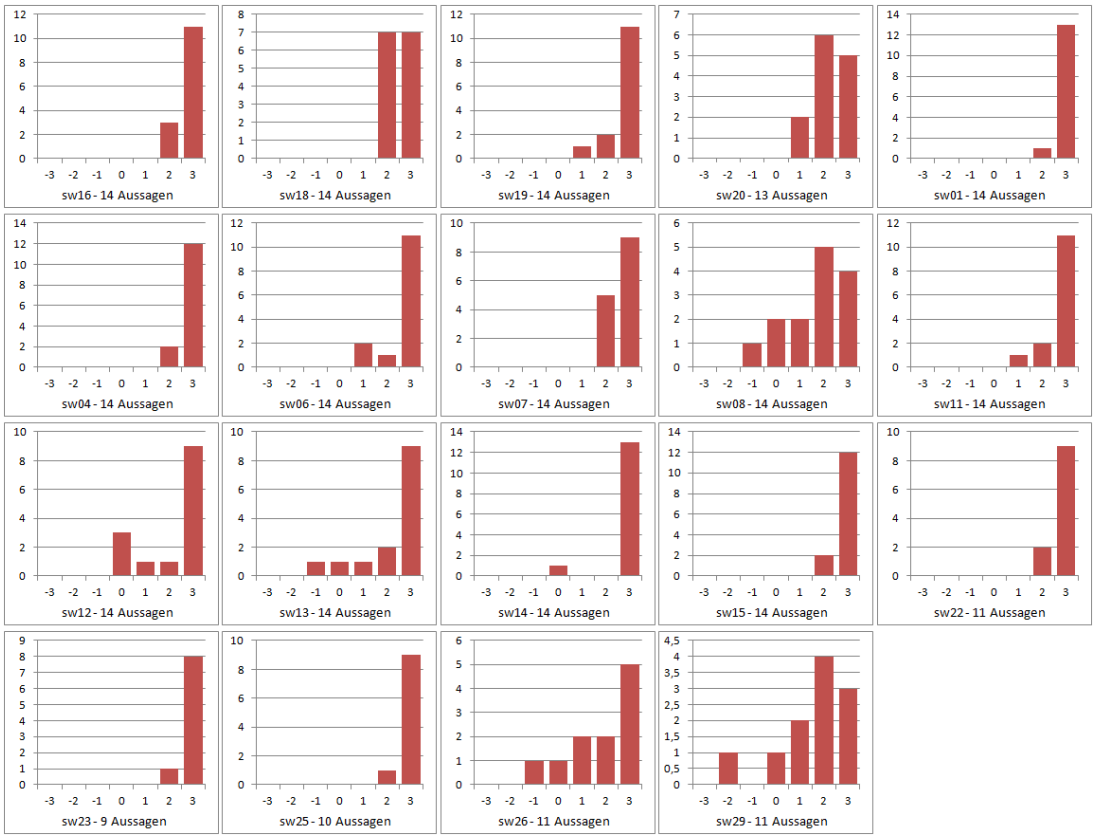


Abbildung 39: Die Darstellung zeigt die über alle Probanden gemittelten Ergebnisse jeder einzelnen Frage.

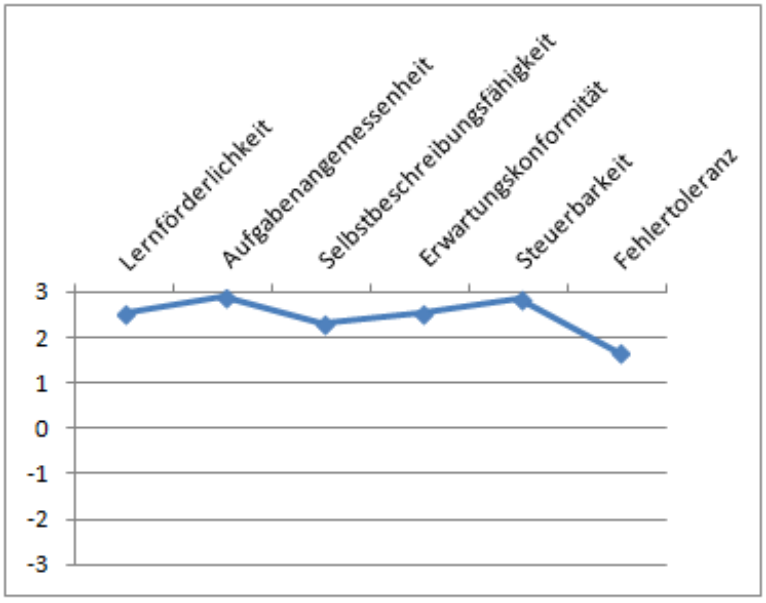


Abbildung 40: Dargestellt sind die Häufigkeiten der Aussagen für jede Frage.

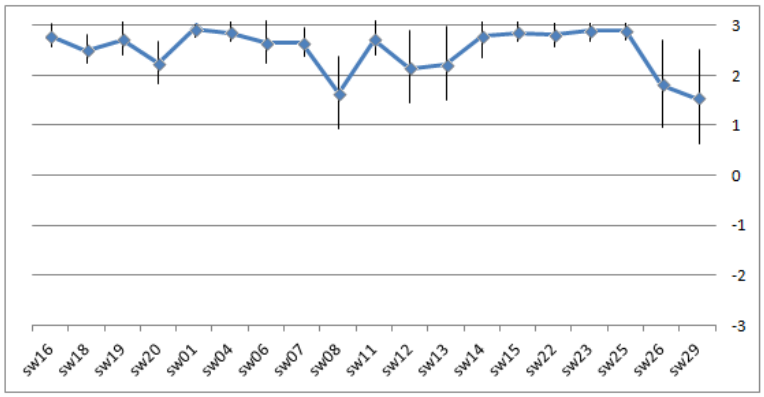


Abbildung 41: Die Darstellung zeigt die gemittelten Werte jeder Frage über alle Probanden. Zusätzlich wird das Konfidenzintervall mit einem Konfidenzniveau von 95% dargestellt.

Hinweis darauf, dass die Anwendung machbare Eingaben nicht ausreichen kommuniziert sowie die Behandlung von Fehlern aufwendig ist. Die positivste Aussage wurde für die Frage sw01 erhalten. Diese beschreibt wie einfach oder kompliziert die Anwendung zu bedienen ist. Das Ergebnis lässt vermuten, dass das Programm im evaluiertem Umfang einfach zu bedienen ist.

Zusätzlich zur Einzelauswertung der Fragen werden nun die Mittelwerte für die einzelnen Usability-Kriterien ermittelt. Die Ergebnisse sind in Abbildung 42 zu sehen. Hierbei sei erneut darauf verwiesen, dass nicht alle Fragen aus jeder Kategorie des Original-Fragebogens übernommen wurden. Daher würden sie sich bspw. nicht für den Vergleich der Kategorien mit den Ergebnissen eines vollständigen Fragebogens eignen. Die am negativsten bewertete Kategorie stellt hierbei die bereits beschriebene *Fehlertoleranz* dar. Aufgrund der negativen Teilbewertung der enthaltenen Fragen ergibt sich dieses Ergebnis. Die am positivsten bewertete Kategorie ist die *Aufgabenangemessenheit*. Dies liefert einen Hinweis darauf, dass die Anwendung ein geeignetes Werkzeug für die Skizzierungsaufgaben darstellt.

5.2.4 *Verbesserung der Anwendung*

Eine Schwierigkeit beim Zeichnen der Probanden betraf das Zeichnen von Aneurysma-Hälsen. Diese Problematik ließe sich durch die von den Probanden gewünschte Funktion lösen, die Gefäßbreite dynamisch einzustellen. Weiterhin kann die Bezeichnung des Werkzeugs von Gefäß- zu *Erstell*-Werkzeug geändert werden. Hierdurch kann eine zum Ausschneide-Werkzeug konsistente Benennung erreicht werden.

Um die anfänglichen Schwierigkeiten der Probanden mit dem *Ausschneide-Werkzeug* zu vermeiden sind zwei Handlungsmöglichkeiten denkbar:

- Die von den Probanden erwartete Funktionalität implementieren und das Werkzeug als *Rasiermesser-* oder *Radierer*-Werkzeug realisieren.
- Die Funktionsweise besser kommunizieren.

Da das Werkzeug, nachdem dessen Bedienung verstanden wurde, positiv bewertet wurde, wird der zweite Ansatz verfolgt. Zur besseren Kommunikation könnten

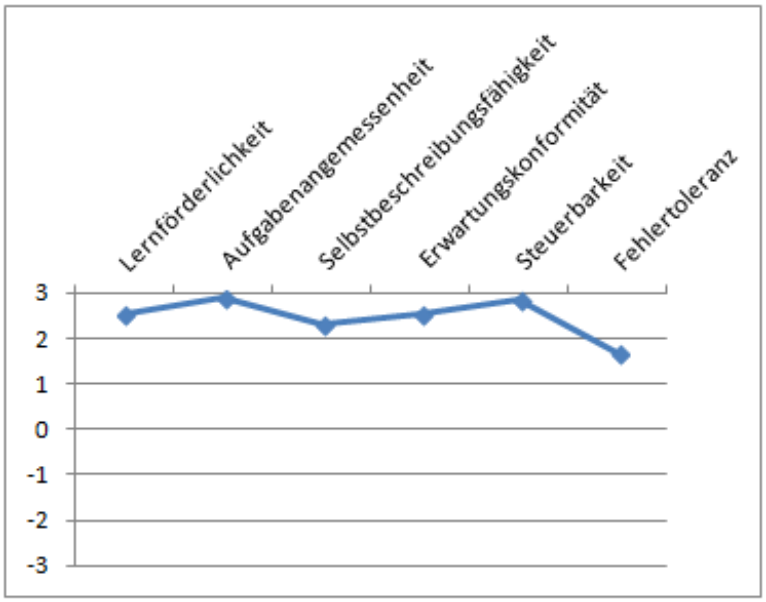


Abbildung 42: Dargestellt ist die gemittelte Bewertung über die einzelnen Usability-Kriterien.

daher (wie auch durch die Probanden empfohlen) Tooltips oder Videos verwendet werden, die die Funktion beschreiben oder aufzeigen.

Das *Coil-Werkzeug* könnte verbessert werden, indem die Erstellung weiter automatisiert wird. Um einen realistischeren Effekt bei der Blutfluss-Simulation zu erhalten könnte die Hindernis-Repräsentation des Coils nicht als durchgezeichnete Linie, sondern als Gestrichelte gerendert werden. Hierdurch könnte Blut bei der Simulation den Coil teilweise durchdringen.

Um das *Stent-Werkzeug* zu verbessern, sollte zunächst eine Fehlermeldung ausgegeben werden, wenn der Stent nicht automatisch platziert werden konnte. Weiterhin könnte dem Nutzer ermöglicht werden, den Stent bei fehlgeschlagener automatischer Platzierung manuell zu platzieren. Zusätzlich sollte das Stent-Werkzeug ausgegraut werden, wenn noch kein Gefäß gezeichnet wurde.

Das *Kraft-Werkzeug* kann verbessert werden, indem die Richtung, in der die Kraft wirkt, umkehrbar ist. Hierdurch kann der Nutzer eine Kraft einzeichnen, ohne die Richtung berücksichtigen zu müssen. Zusätzlich sollte das Symbol des Werkzeugs vereinfacht werden, indem nur noch ein Pfeil dessen Funktion anzeigt. Ein weiterer Punkt zur Verbesserung adressiert die großen Wirkungen, die durch kleine Zeichenungenauigkeiten erzeugt werden. Durch zusätzliches Glätten der gezeichneten Linie können diese gemindert werden. Weiterhin könnte sich der Anfang und das Ende weniger stark auf den Blutfluss auswirken, da dort die größten Ungenauigkeiten beim Zeichnen entstanden.

Eine Verbesserung für das *Tinten-Werkzeug* ist möglich, indem die Benutzerführung an die restlichen Werkzeuge angepasst wird. Hierdurch kann die Erwartungskonformität der Anwendung verbessert werden. Statt also eine punktuelle Farbquelle zu erstellen, kann der Nutzer einen Farbbereich zeichnen. Die Funktionalität, dass verschiedene Farben gezeichnet werden können, kann dem Nutzer durch die Ver-

wendung eines verschiedenfarbigen Symbols für das Werkzeug kommuniziert werden.

Weiterhin kann dem Hinweis aus der Nutzerstudie gefolgt werden und die Werkzeugbereiche *Behandlung* und *Blutfluss* getauscht werden. Durch die Veränderung der Reihenfolge wird der Arbeitsablaufs besser beschrieben.

Von einigen Probanden wurde angemerkt, dass ein zusätzliches, aus anderen Anwendungen bekanntes Werkzeug zum *Aufblasen* und *Entleeren* (engl. inflate/deflate) nützlich wäre. Hiermit könnten einerseits Gefäße *aufgeblasen* werden und somit Aneurysmen erstellt werden. Andererseits könnte durch das *Entleeren* Stenosen und Aneurysmen-Hälse erzeugt werden.

Ein weiterer Vorschlag eines Probanden betraf die Möglichkeit einer Kopierfunktion [P11]. Zunächst wird eine krankhafte Gefäßstruktur erstellt und anschließend der Blutfluss darin visualisiert. Nun wird mithilfe der Kopierfunktion das gezeichnete kopiert um eine Vergleichsvisualisierung für das Einzeichnen der Behandlung zu erhalten.

Die negativsten Ergebnisse bei den Fragebogen wurden im Bereich Fehlertoleranz erhalten. Um die Anwendung in diesem Punkt zu verbessern können Möglichkeiten genutzt werden, um Fehler zu vermeiden (z. B. die bereits beschriebenen Hilfen in Form von Tooltips oder Videos). Weiterhin können dem Nutzer Möglichkeiten gegeben werden, Fehler zu korrigieren. Dies wäre über eine Editierfunktionen möglich oder aber durch eine von den Probanden Empfohlene eine *Rückgängig/Wiederherstellen*-Funktion. Hierdurch würde auch der Usability-Aspekt der Steuerbarkeit adressiert und verbessert werden.

6

ZUSAMMENFASSUNG UND AUSBLICK

In dieser Arbeit wurde zunächst die Eignung vom *Skizzieren* zur Problemlösung herausgestellt und anschließend die Wichtigkeit von *Gefäßerkrankungen* in der Medizin und Volkswirtschaft erläutert. Daraus wurde die Fragestellung entwickelt, wie Skizzen von Gefäßen und deren Erkrankungen mit zusätzlicher Abbildung des Blutflusses der Arzt-Patientenkommunikation helfen können.

Hierzu wurden zunächst die nötigen Grundlagen aus den Bereichen der *Medizin*, der *Strömungslehre* und *skizzenbasierten Benutzungsschnittstellen* vorgestellt. Weiterhin wurden Kriterien identifiziert, die für einen *geeigneten Umgang* mit einer *Benutzungsschnittstelle* relevant sind.

Aus diesen Grundlagen wurde ein Konzept entwickelt, welches das *Skizzieren* von wichtigen Gefäßstrukturen, deren Erkrankungen und Behandlungen ermöglicht und dabei eine echtzeitfähige, integrierte *Simulation von Blutfluss* ermöglicht. Die Entwicklung des Konzeptes berücksichtigte dabei besonders *ergonomische Gesichtspunkte* für die Bedienung.

Basierend auf dem Konzept wurde ein Prototyp entwickelt. Dieser wurde in seiner Grundfunktionalität beschrieben. Zusätzlich wurde exemplarisch auf Besonderheiten der Realisierung eingegangen.

Abschließend wurden geeignete Methoden zur Evaluierung des Prototyps identifiziert. Mit 14 Probanden wurde eine Nutzerstudie durchgeführt, welche qualitative und quantitative Methoden nutzte. Darauf folgte eine Auswertung der Ergebnisse sowie eine Beschreibung von Möglichkeiten den entstandenen Prototyp zu verbessern.

6.1 BEWERTUNG DER ERGEBNISSE

Die Arbeit liefert ein Konzept sowie eine prototypische Umsetzung, mit der Gefäße, Gefäßerkrankungen und Behandlungsmethoden skizziert werden können. In dieser Skizze kann interaktiv und echtzeitfähig Blutfluss simuliert werden, wobei die Simulation unmittelbar auf Änderungen in der Skizze reagiert und durch diese beeinflusst werden kann. Die Eignung dieses Konzeptes wurde durch eine qualitative und quantitative Evaluierung überprüft. Die positiven Ergebnisse der Evaluierung sind ein Indikator dafür, dass das vorgestellte Konzept und die entwickelte Anwendung für das Skizzieren von Gefäßstrukturen geeignet sind. Die Evaluierung zeigt aber auch, dass ein Verbesserungspotential besteht. Das Ergebnis dieser Arbeit kann als Grundlage für die genauere Analyse verwendet werden, wie Methoden zum Skizzieren in einem Arzt-Patienten-Dialog genutzt werden können.

6.2 ZUKÜNFTIGE FRAGESTELLUNGEN

Das Usability-Kriterium der *Individualisierbarkeit* wurde bei der Entwicklung des Konzeptes nicht betrachtet und daher auch nicht durch die Evaluierung überprüft. Für eine geeignete Analyse dieses Kriteriums wäre es nötig zu untersuchen, welche Anpassungen der Anwendung von Ärzten gewünscht werden. Hierfür bietet der entwickelte Prototyp eine geeignete Grundlage.

Eine weitere Fragestellung betrifft die Möglichkeit mit der Anwendung *kollaborativ zu arbeiten*. Hierbei kann genauer untersucht werden, inwieweit mehrere Ärzte die Anwendung gleichzeitig an *verschiedenen Orten* nutzen können. Dabei wären verschiedene Fragestellungen zu untersuchen. Zum einen wie die Kommunikation der Teilnehmer unterstützt werden kann (z.B. durch Sprachübertragung oder Webcam) und zum anderen wie die Nutzerintention eines Anwenders den anderen kommuniziert werden kann. Bei der gleichzeitigen Bedienung sind auch Fragestellungen eines Rechtemanagements relevant. Neben der Zusammenarbeit an verschiedenen Orten könnten auch mehrere Ärzte am selben Ort das Programm als Diskussionswerkzeug nutzen. Zur simultanen Bedienung müsste die Anwendung *Multi-Touch* unterstützen. Neben der gleichzeitigen Verwendung mehrerer Stifte für die Bedienung sind auch verschiedene Eingabemodalitäten denkbar. Mit diesen könnte zusätzlich eine Semantik bei der Eingabe verbunden werden: Mit der Stift-Interaktion werden Objekte *erstellt* und mit der Finger-Interaktion werden sie *editiert*.

Eine interessante Erweiterung für die Anwendung wäre die Verwendung von *Gesten*. Dadurch könnten erstellte Objekte editiert oder Moduswechsel veranlasst werden. Hierbei ist auch zu beachten, dass Gesten gelernt und gemerkt werden müssen. Zusätzlich entstehen also Fragestellungen, die die Kommunikation der möglichen Gesten betreffen.

Die skizzierte Gefäßstruktur könnte in ein *dreidimensionales Modell* umgewandelt werden. Methoden, um aus einer Skizze ein 3D-Modell zu generieren, sind zahlreich vorhanden (z. B. *Teddy* von Igarashi et al. [40]). Das entstandene Modell könnte nun verwendet werden, um den Patienten eine realistischere Vorstellung seiner Gefäßerkrankungen zu bieten. Auf dem entstandenen 3D-Modell kann ebenso eine Blutfluss-Simulation ausgeführt werden. Für eine Prognose wäre ein solches 3D-Modell allerdings nur bedingt geeignet. Dies resultiert daraus, dass die Grundlage eine zweidimensionale Darstellung ist. Die dritte Dimension muss *geschätzt* werden und kann daher kein verlässlicher Prädiktor sein.

Neben der Umwandlung in ein 3D-Modell können zukünftige Arbeiten auch die Fragestellung untersuchen, wie das gesamte Konzept der Arbeit in die dritte Dimension übertragen werden könnte. Dies stellt Teilfragen danach, wie der Blutfluss in 3D *simuliert* und *dargestellt* sowie die Gefäßstruktur in 3D *skizziert* werden kann. Für die Interaktion in 3D gibt es neben der Maus geeignetere Möglichkeiten wie bspw. *zSpace-Display*[1]. Dieser 3D-Monitor erlaubt unter zusätzlicher Verwendung einer 3D-Brille und eines Stiftes zur Eingabe die Interaktion in 3D. Unter Verwendung dieser Hardware und der Simulation des Blutflusses in 3D könnte das vorgestellte Konzept zusätzliche Relevanz in den Anwendungsgebieten der Operationsplanung und dessen Training erhalten.

1 http://zspace.com/, zuletzt aufgerufen: 03.03.2014

LITERATURVERZEICHNIS

[1] Xiaogang Xu A, Wenyin Liu B, Xiangyu Jin A, and Zhengxing Sun A. Sketch-based user interface for creative tasks. In *Proceedings of 5th Asia Pacific Conference on Computer Human Interaction*, page 560–570, 2002.

[2] T. Akenine-Möller, E. Haines, and N. Hoffman. *Real-Time Rendering, Third Edition*. Taylor & Francis, 2011.

[3] Seok-Hyung Bae, Ravin Balakrishnan, and Karan Singh. Ilovesketch: As-natural-as-possible sketching system for creating 3d curve models. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 151–160, New York, NY, USA, 2008. ACM.

[4] Michael Banks and Elaine Cohen. Real time spline curves from interactively sketched data. In *Proceedings of the 1990 Symposium on Interactive 3D Graphics*, I3D '90, pages 99–107, New York, NY, USA, 1990. ACM.

[5] Joshua B Bederson, E Sander Connolly, H Hunt Batjer, Ralph G Dacey, Jacques E Dion, Michael N Diringer, John E Duldner, Robert E Harbaugh, Aman B Patel, and Robert H Rosenwasser. Guidelines for the management of aneurysmal subarachnoid hemorrhage a statement for healthcare professionals from a special writing group of the stroke council, american heart association. *Stroke*, 40(3):994–1025, 1994.

[6] Jennifer L Beebe-Dimmer, John R Pfeifer, Jennifer S Engle, and David Schottenfeld. The epidemiology of chronic venous insufficiency and varicose veins. *Annals of epidemiology*, 15(3):175–184, 2005.

[7] Steven Birr, Volker Dicken, Benjamin Geisler, Konrad Mühler, Bernhard Preim, and Christina Stöcker. Interaktive Reports für die Planung von Lungentumoroperationen. In Maximilian Eibl, editor, *Mensch & Computer 2011: überMEDIEN | ÜBERmorgen*, pages 131–140, München, 2011. Oldenbourg Verlag.

[8] D. Borland and R.M. Taylor. Rainbow color map (still) considered harmful. *Computer Graphics and Applications, IEEE*, 27(2):14–17, 2007.

[9] Robert Bridson. *Fluid Simulation for Computer Graphics*. A K Peters/CRC Press, September 2008.

[10] J. C. Butcher. A history of runge-kutta methods. *Appl. Numer. Math.*, 20(3): 247–260, March 1996.

[11] Bill Buxton. *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann, first edition, March 2007.

[12] Brian Cabral and Leith Casey Leedom. Imaging vector fields using line integral convolution. In *Proceedings of the 20th Annual Conference on Computer Graphics*

and Interactive Techniques, SIGGRAPH '93, pages 263–270, New York, NY, USA, 1993. ACM.

[13] S.K. Card, J.D. Mackinlay, and B. Shneiderman. *Readings in Information Visualization: Using Vision to Think*. Interactive Technologies Series. Morgan Kaufmann Publishers, 1999.

[14] A.J. Chorin and J.E. Marsden. *A Mathematical Introduction to Fluid Mechanics*. Texts in Applied Mathematics. Springer, 1993.

[15] P.G. Ciarlet, R. Glowinski, and J.L. Lions. *Numerical Methods for Non-Newtonian Fluids: Special Volume*. Handbook of Numerical Analysis Handbook of Numerical Analysis. Elsevier, 2011.

[16] J.S. Cohen and I. Stewart. *The Collapse of Chaos: Discovering Simplicity in a Complex World*. Penguin Science Series. Penguin Books, Limited, 1995.

[17] Matthew T. Cook and Arvin Agah. A survey of sketch-based 3-d modeling techniques. *Interact. Comput.*, 21(3):201–211, July 2009.

[18] Willem C. de Leeuw and Jarke J. van Wijk. A probe for local flow field visualization. In *Proceedings of the 4th Conference on Visualization '93*, VIS '93, pages 39–45, Washington, DC, USA, 1993. IEEE Computer Society.

[19] H.-H. Eckstein and A. Zimmermann. Gefäße. In *Basiswissen Chirurgie*, Springer-Lehrbuch, pages 195–221. Springer Berlin Heidelberg, 2010.

[20] Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. Visual simulation of smoke. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 15–22, New York, NY, USA, 2001. ACM.

[21] Joel H. Ferziger and Milovan Perić. Compressible flow. In *Computational Methods for Fluid Dynamics*, pages 309–328. Springer Berlin Heidelberg, 2002.

[22] Lisa K. Forssell. Visualizing flow over curvilinear grid surfaces using line integral convolution. In *Proceedings of the Conference on Visualization '94*, VIS '94, pages 240–247, Los Alamitos, CA, USA, 1994. IEEE Computer Society Press.

[23] Michael Frank. *Wege in der Physikdidaktik, Band 2 - Anregungen für Unterricht und Lehre*. Verlag Palm und Enke, Erlangen, 1991.

[24] Juhana Frösen, Riikka Tulamo, Anders Paetau, Elisa Laaksamo, Miikka Korja, Aki Laakso, Mika Niemelä, and Juha Hernesniemi. Saccular intracranial aneurysm: pathology and mechanisms. *Acta Neuropathologica*, 123(6):773–786, 2012. ISSN 0001-6322.

[25] Rocco Gasteiger. *Visual Exploration of Cardiovascular Hemodynamics*. PhD thesis, Otto-von-Guericke-Universität Magdeburg, 2013. Wird gedruckt.

[26] Rocco Gasteiger, Mathias Neugebauer, Volker Diehl, Oliver Beuing, and Bernhard Preim. Entwurf einer angepassten Visualisierung von zerebralen Aneurysmen mit innenliegenden Blutflussinformationen. In *Proc. of CURAC*, pages 61–66, Düsseldorf, November 2010.

[27] Sylvia Glaßer, Kai Lawonn, and Bernhard Preim. Visualization of 3D Cluster Results for Medical Tomographic Image Data. In *In Proc. of Conference on Computer Graphics Theory and Applications (VISIGRAPP/GRAPP)*, pages 169–176, 2014.

[28] Gonçalo Amador Abel Gomes. Linear solvers for stable fluids: Gpu vs cpu. 2009.

[29] François Guimbretiére and Terry Winograd. Flowmenu: Combining command, text, and data entry. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, UIST '00, pages 213–216, New York, NY, USA, 2000. ACM.

[30] C.D. Hansen and C.R. Johnson. *The Visualization Handbook*. Referex Engineering. Butterworth-Heinemann, 2005.

[31] Francis H. Harlow and J. Eddie Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Physics of Fluids*, 8(12):2182–2189, 1965.

[32] Mark J. Harris. *GPU GEMS Chapter 38, Fast Fluid Dynamics Simulation on the GPU*. Pearson Higher Education, 2004.

[33] Marc Hassenzahl, Michael Burmester, and Franz Koller. Attrakdiff: Ein fragebogen zur messung wahrgenommener hedonischer und pragmatischer qualität. In Gerd Szwillus and Jürgen Ziegler, editors, *Mensch & Computer 2003*, volume 57 of *Berichte des German Chapter of the ACM*, pages 187–196. Viweg+Teubner Verlag, 2003.

[34] Frank Heckel, Jan H. Moltz, Christian Tietjen, and Horst K. Hahn. Sketch-based editing tools for tumour segmentation in 3d medical images. *Computer Graphics Forum*, 32(8):144–157, 2013.

[35] J. Herold and T. F. Stahovich. The 1¢ recognizer: A fast, accurate, and easy-to-implement handwritten gesture recognition technique. In *Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling*, SBIM '12, pages 39–46, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association.

[36] Donovan Hohn. *Moby-Duck: The True Story of 28,800 Bath Toys Lost at Sea*. Union, London, 2012.

[37] Andreas Holzinger. Usability engineering methods for software developers. *Communications of the ACM*, 48(1):71–74, 2005.

[38] J. B. Hurst, M. S. Mhoney, Norman H. Taylor, D. T. Ross, R. M. Fano, Jr John T. Gilmore, Lawrence G. Roberts, and A. Robin Forrest. Retrospectives ii: the early years in computer graphics at mit, lincoln lab, andd harvard. In *ACM SIGGRAPH 89 Panel Proceedings*, pages 39–73, 1989. http://www.odysci.com/article/1010112994920377.

[39] Takeo Igarashi, Satoshi Matsuoka, Sachiko Kawachiya, and Hidehiko Tanaka. Interactive beautification: A technique for rapid geometric design. In *ACM SIGGRAPH 2007 Courses*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[40] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: A sketching interface for 3d freeform design. In *ACM SIGGRAPH 2007 Courses*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[41] Robert J.K. Jacob, Audrey Girouard, Leanne M. Hirshfield, Michael S. Horn, Orit Shaer, Erin Treacy Solovey, and Jamie Zigelbaum. Reality-based interaction: A framework for post-wimp interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 201–210, New York, NY, USA, 2008. ACM.

[42] Bruno Jobard and Wilfrid Lefer. Creating evenly-spaced streamlines of arbitrary density. In Wilfrid Lefer and Michel Grave, editors, *Visualization in Scientific Computing '97*, Eurographics, pages 43–55. Springer Vienna, 1997.

[43] Bruno Jobard and Wilfrid Lefer. Unsteady flow visualization by animating evenly-spaced streamlines. *Comput. Graph. Forum*, 19(3):31–39, 2000.

[44] Joaquim Jorge and Faramarz Samavati. *Sketch-based Interfaces and Modeling*. Springer London, 2011.

[45] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *INTERNATIONAL JOURNAL OF COMPUTER VISION*, 1(4):321–331, 1988.

[46] B.J. Keulers. *Computer-based Patient Education: Its Potential in General and Plastic Surgery*. PhD thesis, 2008.

[47] O. Kreylos, A. M. Tesdall, B. Hamann, J. K. Hunter, and K. I. Joy. Interactive visualization and steering of cfd simulations. In *Proceedings of the Symposium on Data Visualisation 2002*, VISSYM '02, pages 25–34, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association.

[48] Jens Krüger and Rüdiger Westermann. Linear algebra operators for gpu implementation of numerical algorithms. *ACM Trans. Graph.*, 22(3):908–916, July 2003.

[49] Gordon Kurtenbach and William Buxton. User learning and performance with marking menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, pages 258–264, New York, NY, USA, 1994. ACM.

[50] Robert S. Laramee, Helwig Hauser, Helmut Doleisch, Benjamin Vrolijk, Frits H. Post, and Daniel Weiskopf. The state of the art in flow visualization: Dense and texture-based techniques. *Computer Graphics Forum*, 23:2004, 2004.

[51] Kai Lawonn, Tobias Moench, and Bernhard Preim. Streamlines for illustrative real-time rendering. *Computer Graphics Forum*, 32(3pt3):321–330, 2013.

[52] Kai Lawonn, Patrick Saalfeld, and Bernhard Preim. Illustrative Visualization of Endoscopic Views. In *Bildverarbeitung für die Medizin (BVM)*, page in print, 2014.

[53] Endre M. Lidal. *Sketch-based Storytelling for Cognitive Problem Solving*. PhD thesis, Department of Informatics, University of Bergen, Norway, June 2013.

[54] A. Light and P.J. Bartlein. The end of the rainbow? color schemes for improved data graphics. *EOS Transactions of the American Geophysical Union*, 85(40):385, 2004.

[55] M. B. Liu, G. R. Liu, and Z. Zong. An overview on smoothed particle hydrodynamics. *International Journal of Computational Methods*, 05(01):135–188, 2008.

[56] E. MacCurdy. *The Notebooks of Leonardo Da Vinci*. Braziller, 1955.

[57] Lev. Manovich. *The language of new media*. MIT Press, Cambridge, Mass., 2002.

[58] Diane Marinkas, Robert C. Zeleznik, and Joseph J. LaViola, Jr. Shadow buttons: Exposing wimp functionality while preserving the inking surface in sketch-based interfaces. In *Proceedings of the 6th Eurographics Symposium on Sketch-Based Interfaces and Modeling*, SBIM '09, pages 159–164, New York, NY, USA, 2009. ACM.

[59] Ajith Mascarenhas and Jack Snoeyink. Isocontour based visualization of time-varying scalar fields. In Torsten Möller, Bernd Hamann, and RobertD. Russell, editors, *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, Mathematics and Visualization, pages 41–68. Springer Berlin Heidelberg, 2009.

[60] Tony McLoughlin, Robert S. Laramee, Ronald Peikert, Frits H. Post, and Min Chen. Over two decades of integration-based, geometric flow visualization. *Computer Graphics Forum*, 29:1807–1829, 2010.

[61] S. Mendis, P. Puska, B. Norrving, World Health Organization, World Heart Federation, and World Stroke Organization. *Global Atlas on Cardiovascular Disease Prevention and Control*. Nonserial Publications Series. World Health Organization in collaboration with the World Heart Federation and the World Stroke Organization, 2011.

[62] Matthias Müller, Simon Schirm, and Matthias Teschner. Interactive blood simulation for virtual surgery based on smoothed particle hydrodynamics. *Technol. Health Care*, 12(1):25–31, February 2004.

[63] Ferran Naya, Manuel Contero, Nuria Aleixos, and Pedro Company. Parsketch: A sketch-based interface for a 2d parametric geometry editor. In *Proceedings of the 12th International Conference on Human-computer Interaction: Interaction Platforms and Techniques*, HCI'07, pages 115–124, Berlin, Heidelberg, 2007. Springer-Verlag.

[64] Mathias Neugebauer, Volker Diehl, Martin Skalej, and Bernhard Preim. Geometric Reconstruction of the Ostium of Cerebral Aneurysms. In Reinhard Koch, Andreas Kolb, and Christopf Rezk-Salama, editors, *VMV 2010 - Vision, Modeling, Visualization*, pages 307–314, Siegen, 15.-17. November 2010.

[65] Mathias Neugebauer, Kai Lawonn, Oliver Beuing, and Bernhard Preim. Automatic Generation of Anatomic Characteristics from Cerebral Aneurysm Surface Models. *International Journal of Computer Assisted Radiology and Surgery*, 8(2):279–289, March 2013.

[66] W. Nitsche. Strömungssichtbarmachung. In *Strömungsmesstechnik*, VDI-Buch, pages 154–176. Springer Berlin Heidelberg, 2006.

[67] L. Olsen, F. F. Samavati, M. Costa Sousa, and J. Jorge. A taxonomy of modeling techniques using sketch-based interfaces. In Theoharis Theoharis and Philip Dutré, editors, *Eurographics 2008 - State of the Art Reports (STARs)*, pages 39–57, Crete, Greece, Apr 14-18 2008. Eurographics Association.

[68] Luke Olsen and Faramarz F. Samavati. Image-assisted modeling from sketches. In *Proceedings of Graphics Interface 2010*, GI '10, pages 225–232, Toronto, Ont., Canada, Canada, 2010. Canadian Information Processing Society.

[69] Stanley Osher and Ronald P. Fedkiw. Level set methods: An overview and some recent results. *J. Comput. Phys*, 169:463–502, 2001.

[70] L. Piegl. Interactive data interpolation by rational bezier curves. *IEEE Comput. Graph. Appl.*, 7(4):45–58, April 1987.

[71] Adeline Pihuit, Marie-Paule Cani, and Olivier Palombi. Sketch-based modeling of vascular systems: A first step towards interactive teaching of anatomy. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium*, SBIM '10, pages 151–158, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.

[72] F Post, B Vrolijk, H Hauser, R Laramee, and H Doleisch. The state of the art in flow visualization: Feature extraction and tracking. *Published in Computer Graphics Forum*, 4(22):pp. 775–792, 2003.

[73] Frits H. Post and Jarke J. van Wijk. Visual representation of vector fields: Recent developments and research directions. In *SCIENTIFIC VISUALIZATION – ADVANCES AND CHALLENGES*, pages 367–390. Academic Press, 1994.

[74] Emil Praun, Hugues Hoppe, Matthew Webb, and Adam Finkelstein. Real-time hatching. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 581–, New York, NY, USA, 2001. ACM.

[75] Bernhard. Preim and Raimund. Dachselt. *Interaktive Systeme. Band 1: Grundlagen, Graphical User Interfaces, Informationsvisualisierung*. Springer, Berlin, 2010.

[76] Christian Prinz and Ilka Ott. Gefäße. In *Basiswissen Innere Medizin*, Springer-Lehrbuch, pages 28–51. Springer Berlin Heidelberg, 2012.

[77] Jing Qin, Wai-Man Pang, Binh P. Nguyen, Dong Ni, and Chee-Kong Chui. Particle-based simulation of blood flow and vessel wall interactions in virtual surgery. In *Proceedings of the 2010 Symposium on Information and Communication Technology*, SoICT '10, pages 128–133, New York, NY, USA, 2010. ACM.

[78] B. Reiner and E. Siegel. Radiology reporting: returning to our image-centric roots. *American Journal of Roentgenology*, 187(5):1151–1155, 2006.

[79] A. Sanna, P. Montuschi, and P. Montuschi. A survey on visualization of vector fields by texture-based methods. *Devel. Pattern Rec*, 1:2000, 2000.

[80] D. Schroeder, D. Coffey, and D. Keefe. Drawing with the flow: A sketch-based interface for illustrative visualization of 2d vector fields. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium*, SBIM '10, pages 49–56, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.

[81] Tevfik Metin Sezgin and All Davis. Scale-space based feature point detection for digital ink. In *In AAAI 2004 Fall Symposium Series, Sketch Understanding*, 2004.

[82] Tevfik Metin Sezgin, Thomas Stahovich, and Randall Davis. Sketch based interfaces: Early processing for sketch understanding. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[83] Wei Shen and Alex Pang. Tuft flow visualization. *Proceedings of the Second IASTED International Conference on Visualization, Image, and Image Processing*, 1: 705–710, 2002.

[84] Ben Shneiderman. The future of interactive systems and the emergence of direct manipulation. *Behaviour & Information Technology*, 1:237 – 256, 1982.

[85] Philippe St-Germain, Ioan Nistor, and Ronald Townsend. Numerical modeling of the impact with structures of tsunami bores propagating on dry and wet beds using the sph method. *International Journal of Protective Structures*, 3:221–256, 2012.

[86] Detlev Stalling and Hans-Christian Hege. Fast and resolution independent line integral convolution. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pages 249–256, New York, NY, USA, 1995. ACM.

[87] Jos Stam. Stable fluids. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 121–128, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[88] Andreas Sundquist. Dynamic line integral convolution for visualizing streamline evolution. *IEEE Trans. Vis. Comput. Graph.*, 9(3):273–282, 2003.

[89] Ivan E. Sutherland. *Sketchpad, A Man-Machine Graphical Communication System*. Outstanding Dissertations in the Computer Sciences. Garland Publishing, New York, 1963.

[90] Ivan Edward Sutherland. Sketchpad: A man-machine graphical communication system. Technical Report UCAM-CL-TR-574, University of Cambridge, Computer Laboratory, September 2003.

[91] G. Taubin. Curve and surface smoothing without shrinkage. In *Proceedings of the Fifth International Conference on Computer Vision*, ICCV '95, pages 852–, Washington, DC, USA, 1995. IEEE Computer Society.

[92] Christian Teitzel, Roberto Grosso, and Thomas Ertl. Efficient and reliable integration methods for particle tracing in unsteady flows on discrete meshes. In Wilfrid Lefer and Michel Grave, editors, *Visualization in Scientific Computing '97*, Eurographics, pages 31–41. Springer Vienna, 1997.

[93] Steve Tsang, Ravin Balakrishnan, Karan Singh, and Abhishek Ranjan. A suggestive interface for image guided 3d sketching. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 591–598, New York, NY, USA, 2004. ACM.

[94] Greg Turk and David Banks. Image-guided streamline placement. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 453–460, New York, NY, USA, 1996. ACM.

[95] Andries van Dam. Post-wimp user interfaces. *Commun. ACM*, 40(2):63–67, February 1997.

[96] Daniel Weiskopf. Vector field visualization. In *GPU-Based Interactive Visualization Techniques*, pages 81–159. Springer Berlin Heidelberg, 2007.

[97] Cathleen Wharton, John Rieman, Clayton Lewis, and Peter Polson. The cognitive walkthrough method: A practitioner's guide. In Jakob Nielsen and Robert L. Mack, editors, *Usability Inspection Methods*, pages 105–140. John Wiley & Sons, Inc., New York, NY, USA, 1994.

[98] Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, pages 159–168, New York, NY, USA, 2007. ACM.

[99] George K.C. Wong, Hai-Bin Tan, Marco C.L. Kwan, Rebecca Y.T. Ng, Simon C.H. Yu, Xian-Lun Zhu, and Wai-Sang Poon. Evolution of intracranial aneurysm treatment: From hunterian ligation to the flow diverter. *Surgical Practice*, 15(1):16–20, 2011.

[100] Enhua Wu, Youquan Liu, and Xuehui Liu. An improved study of real-time fluid simulation on gpu: Research articles. *Comput. Animat. Virtual Worlds*, 15 (3-4):139–146, July 2004.

[101] Bo Zhu, Michiaki Iwata, Ryo Haraguchi, Takashi Ashihara, Nobuyuki Umetani, Takeo Igarashi, and Kazuo Nakazawa. Sketch-based dynamic illustration of fluid systems. In *Proceedings of the 2011 SIGGRAPH Asia Conference*, SA '11, pages 134:1–134:8, New York, NY, USA, 2011. ACM.

KOLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

http://code.google.com/p/classicthesis/

Happy users of `classicthesis` usually send a real postcard to the author, a collection of postcards received so far is featured here:

http://postcards.miede.de/

Final Version as of 4. September 2015 (`classicthesis` version 0.1).

SELBSTSTÄNDIGKEITSERKLÄRUNG

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbstständig und nur unter Zuhilfenahme der angegebenen Quellen erstellt habe.

Magdeburg, den 11.März 2014

Patrick Saalfeld

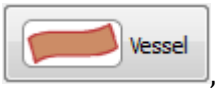
A

ANHANG

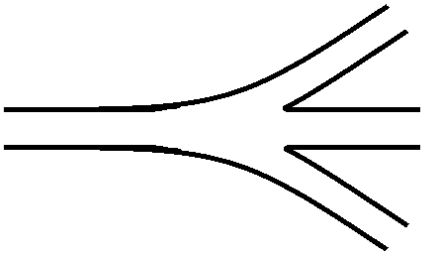
Im Anhang befinden sich:

- Aufgabenblatt der Nutzerstudie
- Fragebogen der Nutzerstudie

ÜBUNG:

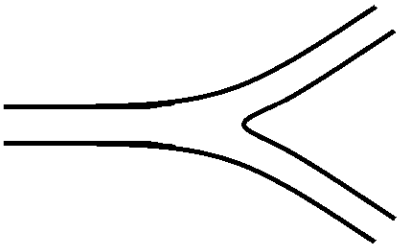
Übung 1: Nutzen Sie das *Vessel*-Werkzeug 

, um die folgende Gefäßstruktur zu zeichnen:



Übung 2: Nutzen sie nun zusätzlich das *Cut*-Werkzeug 

, um die folgende Gefäßstruktur zu zeichnen:



AUFGABEN:

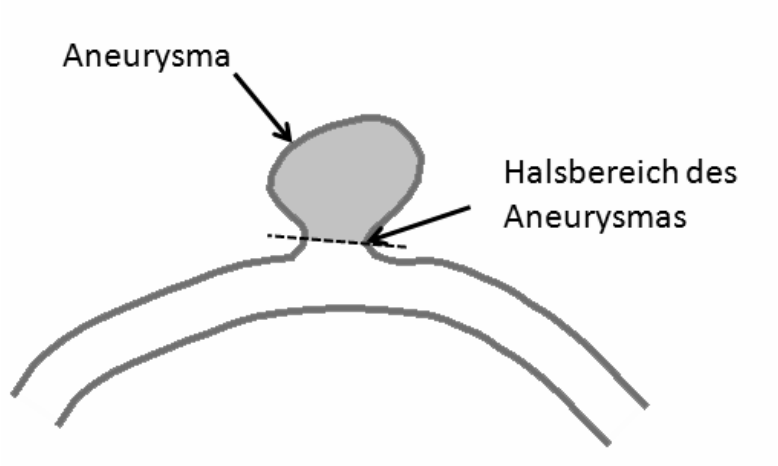
Aufgabe 1: Zeichnen Sie die vorliegende Gefäßverengung mithilfe des *Vessel*-Wekzeugs 

und des *Cut*-Werkzeugs 

.

Aufgabe 2: Zeichnen sie zunächst die vorliegende Gefäßstruktur und nutzen sie anschließend das *Clipping*-Werkzeug 

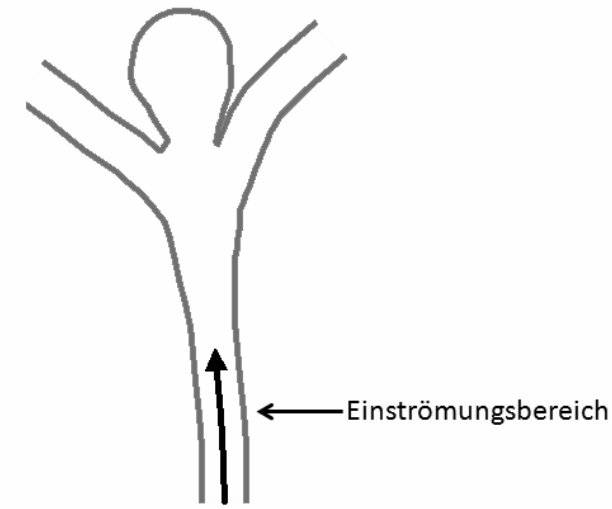
, um im Halsbereich des Aneurysmas einen Clip einzuzeichnen.



Aufgabe 3a: Zeichnen sie das vorliegende Gefäß. Nutzen sie anschließend das *Direction*-Werkzeug 

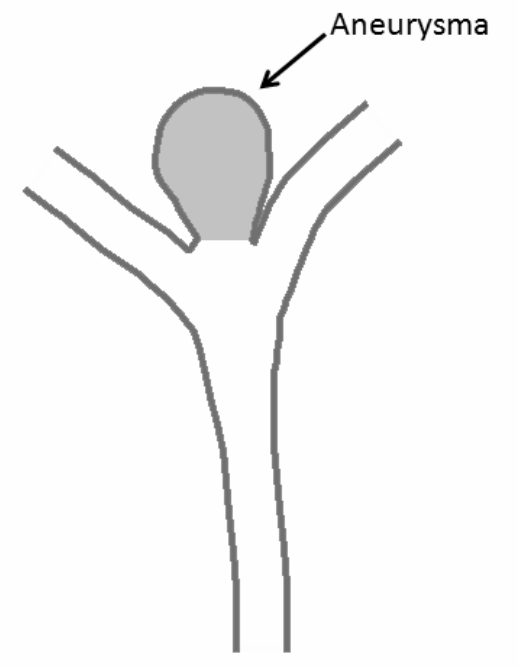
, um den Blutfluss im Einströmungsbereich wie in der Abbildung dargestellt einzuzeichnen. Visualisieren sie den Blutfluss, indem sie mithilfe des *Dye*-Werkzeugs 

Blut im Gefäß platzieren.



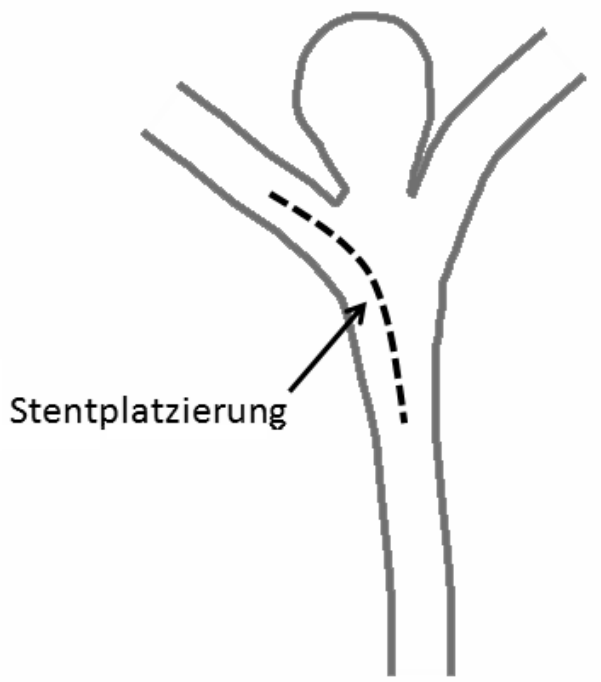
Aufgabe 3b: Nutzen sie nun das *Coiling*-Werkzeug 

, um den im folgenden Bild markierten Bereich des Aneurysmas auszufüllen.



Aufgabe 3c: Platzieren sie nun mithilfe des *Stent*-Werkzeugs 

einen Stent im linken Gefäßast.



Usability-Studie in Anlehnunng an ISONORM 9241/110L zur Masterarbeit „Skizzieren von Gefäßen und Gefäßerkrankungen mit integriertem Blutfluss“

Hinweis: Die Auswertung der Daten erfolgt völlig anonym und dient lediglich zu Forschungszwecken.

Allgemeine Angaben:

Alter: ___ Jahre

Geschlecht: ☐ weiblich ☐ männlich

Erfahrung mit der Nutzung von Computern: seit ___ Jahren

Erfahrung mit Stiftinteraktion am Computer: ☐ ja ☐ nein

Lernförderlichkeit

Ist die Software so gestaltet, dass Sie sich gut darin einarbeiten konnten und bietet sie auch dann Unterstützung, wenn Sie neue Funktionen lernen möchten?

	Die Software ...	- - -	- -	-	-/+	+	++	+++	*Die Software ...*
sw16	erfordert viel Zeit zum Erlernen.	○	○	○	○	○	○	○	erfordert wenig Zeit zum Erlernen.
sw18	erfordert, dass man sich viele Details merken muss.	○	○	○	○	○	○	○	erfordert nicht, dass man sich viele Details merken muss.
sw19	ist so gestaltet, dass sich einmal Gelerntes schlecht einprägt.	○	○	○	○	○	○	○	ist so gestaltet, dass sich einmal Gelerntes gut einprägt.
sw20	ist schlecht ohne fremde Hilfe oder Handbuch erlernbar.	○	○	○	○	○	○	○	ist gut ohne fremde Hilfe oder Handbuch erlernbar.

Bitte beschreiben Sie ein Beispiel, welches die Verletzung der oben genannten Aspekte gegebenenfalls besonders deutlich veranschaulicht:

_____

_____

_____

Aufgabenangemessenheit

Unterstützt die Software die Erledigung Ihrer Arbeitsaufgaben, ohne Sie unnötig zu belasten?

	Die Software ...	- - -	- -	-	-/+	+	++	+++	*Die Software ...*
sw01	ist kompliziert zu bedienen.	○	○	○	○	○	○	○	ist unkompliziert zu bedienen.
sw04	erfordert überflüssige Eingaben.	○	○	○	○	○	○	○	erfordert keine überflüssigen Eingaben.

Bitte beschreiben Sie ein Beispiel, welches die Verletzung der oben genannten Aspekte gegebenenfalls besonders deutlich veranschaulicht:

_____

_____

_____

Selbstbeschreibungsfähigkeit

Gibt Ihnen die Software genügend Erläuterungen und ist sie in ausreichendem Maße verständlich?

	Die Software ...	- - -	- -	-	-/+	+	++	+++	*Die Software ...*
sw06	bietet einen schlechten Überblick über ihr Funktionsangebot.	○	○	○	○	○	○	○	bietet einen guten Überblick über ihr Funktionsangebot.
sw07	verwendet schlecht verständliche Begriffe, Bezeichnungen, Abkürzungen oder Symbole in Masken und Menüs.	○	○	○	○	○	○	○	verwendet gut verständliche Begriffe, Bezeichnungen, Abkürzungen oder Symbole in Masken und Menüs.
sw08	liefert in unzureichendem Maße Informationen darüber, welche Eingaben zulässig oder nötig sind.	○	○	○	○	○	○	○	liefert in zureichendem Maße Informationen darüber, welche Eingaben zulässig oder nötig sind.

Bitte beschreiben Sie ein Beispiel, welches die Verletzung der oben genannten Aspekte gegebenenfalls besonders deutlich veranschaulicht:

_____

_____

_____

Erwartungskonformität

Kommt die Software durch eine einheitliche und verständliche Gestaltung Ihren Erwartungen und Gewohnheiten entgegen?

	Die Software ...	- - -	- -	-	-/+	+	++	+++	*Die Software ...*
sw11	erschwert die Orientierung durch eine uneinheitliche Gestaltung.	○	○	○	○	○	○	○	erleichtert die Orientierung durch eine einheitliche Gestaltung.
sw12	lässt einen im Unklaren darüber, ob eine Eingabe erfolgreich war oder nicht.	○	○	○	○	○	○	○	lässt einen nicht im Unklaren darüber, ob eine Eingabe erfolgreich war oder nicht.
sw13	informiert in unzureichendem Maße über das, was es gerade macht.	○	○	○	○	○	○	○	informiert in ausreichendem Maße über das, was es gerade macht.
sw14	reagiert mit schwer vorhersehbaren Bearbeitungszeiten.	○	○	○	○	○	○	○	reagiert mit gut vorhersehbaren Bearbeitungszeiten.
sw15	lässt sich nicht durchgehend nach einem einheitlichen Prinzip bedienen.	○	○	○	○	○	○	○	lässt sich durchgehend nach einem einheitlichen Prinzip bedienen.

Bitte beschreiben Sie ein Beispiel, welches die Verletzung der oben genannten Aspekte gegebenenfalls besonders deutlich veranschaulicht:

_____

_____

_____

Steuerbarkeit

Können Sie die Art und Weise, wie Sie mit der Software arbeiten, beeinflussen?

	Die Software ...	- - -	- -	-	-/+	+	++	+++	*Die Software ...*
sw22	erzwingt eine unnötig starre Einhaltung von Bearbeitungsschritten.	○	○	○	○	○	○	○	erzwingt keine unnötig starre Einhaltung von Bearbeitungsschritten.
sw23	ermöglicht keinen leichten Wechsel zwischen einzelnen Menüs oder Masken.	○	○	○	○	○	○	○	ermöglicht einen leichten Wechsel zwischen einzelnen Menüs oder Masken.
sw25	erzwingt unnötige Unterbrechungen der Arbeit.	○	○	○	○	○	○	○	erzwingt keine unnötigen Unterbrechungen der Arbeit.

Bitte beschreiben Sie ein Beispiel, welches die Verletzung der oben genannten Aspekte gegebenenfalls besonders deutlich veranschaulicht:

_____

_____

_____

Fehlertoleranz

Bietet Ihnen die Software die Möglichkeit, trotz fehlerhafter Eingaben das beabsichtigte Arbeitsergebnis ohne oder mit geringem Korrekturaufwand zu erreichen?

	Die Software ...	- - -	- -	-	-/+	+	++	+++	*Die Software ...*
sw26	ist so gestaltet, dass kleine Fehler schwerwiegende Folgen haben können.	○	○	○	○	○	○	○	ist so gestaltet, dass kleine Fehler keine schwerwiegenden Folgen haben können.
sw29	erfordert bei Fehlern im Großen und Ganzen einen hohen Korrekturaufwand.	○	○	○	○	○	○	○	erfordert bei Fehlern im Großen und Ganzen einen geringen Korrekturaufwand.

Bitte beschreiben Sie ein Beispiel, welches die Verletzung der oben genannten Aspekte gegebenenfalls besonders deutlich veranschaulicht:

_____

_____

_____

Vielen Dank für die Teilnahme an dieser Studie!