Visualisierungsinstitut der Universität Stuttgart
Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Diplomarbeit Nr. 3570

Überlagerung von charakteristischen Linien

Sebastian Konle

Studiengang:	Informatik
Prüfer:	Prof. Dr. Daniel Weiskopf
Betreuer:	Dipl.-Inf. Marcel Hlawatsch M.Sc. Grzegorz Karch
begonnen am:	1. Oktober 2013
beendet am:	2. April 2014
CR-Klassifikation:	I.3.3, I.3.m, J.2, I.6.6

Kurzfassung

Diese Arbeit befasst sich mit der Problematik, dass bei der Verwendung von Pfad- oder Streichlinien um instationäre Strömungen zu beschreiben Überlagerungen entstehen können und somit Informationen verschleiert werden. Dabei sind die Strömungen als zweidimensionale zeitabhängige Vektorfelder vorgegeben.

Hierfür wird untersucht ob durch das Einbringen geeigneter Überlagerungstechniken der Informationsgehalt des erzeugten Strömungsbildes signifikant erhöht werden kann.

Die Erzeugung der hierfür verwendeten Pfad- und Streichlinien werden auf der GPU durchgeführt, um die Berechnungszeit durch Parallelisieren zu verkürzen. Zudem wird durch mehrstufiges Zeichnen der Speicheraufwand minimiert.

Die durch verschiedene Saattechniken gewonnenen Strömungsbilder werden verglichen und es wird untersucht aus welchem Grund die erkennbaren Strukturen entstanden sind.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Strom- Streich- und Bahnlinien sind charakteristische Linien, mit denen eine Strömung beschrieben werden kann. Wobei Streich- und Bahnlinien in einer stationären (zeitlich konstanten) Strömung den gleichen Verlauf haben wie Stromlinien. Stromlinien beschreiben daher stets stationäre Strömung. Je komplexer eine Strömung ist, umso komplizierter wird es sie mit diesen Linien vollständig zu beschreiben, da zu dicht liegende oder sich überlagernde Linien es erschweren oder gar unmöglich machen den Strömungsverlauf darzustellen. Verwendet man hingegen zu wenige Linien, können wichtige Strukturen der Strömung nicht dargestellt werden. In Abbildung 1.1 wird dies veranschaulicht indem Strömungsbilder mit unterschiedlich vielen Bahnlinien miteinander verglichen werden.

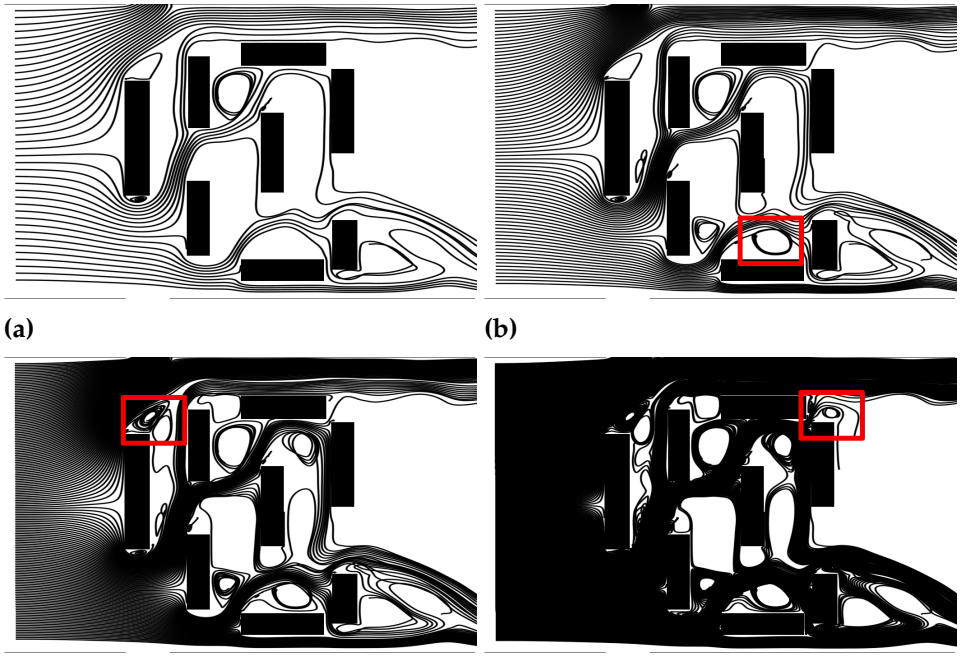
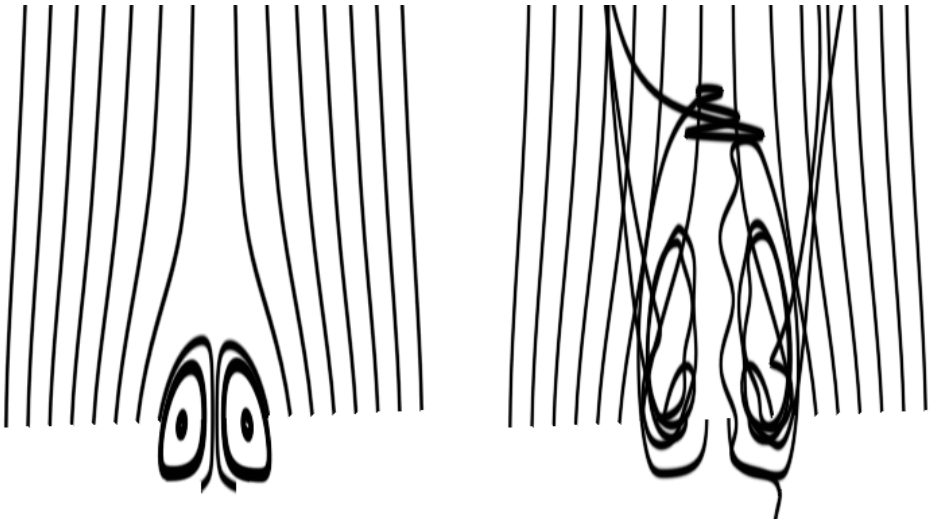


Abbildung 1.1: In Bild (a) wurden 40, in (b) 80, in (c) 160 und in Bild (b) 640 Bahnlinien entlang einer Linie gesät. Man sieht, dass bei aufsteigender Saatdichte neue Strömungsdetails sichtbar werden, andere Details durch Überlagerung jedoch verloren gehen. Durch die roten Rechtecke wurden einige der neu hinzugekommenen Strömungsdetails hervorgehoben.

Um das Problem der Überlagerung zu umgehen, können bei stationären Strömungen verschiedene Techniken angewendet werden. Einige der Techniken werden in Kapitel 2 kurz beschrieben. Dies ist möglich da sich Stromlinien, die den Strömungsverlauf darstellen, nicht schneiden können.

Bei instationären Strömungen, also Strömungen die sich zeitlich ändern, können sich die Linien kreuzen, wodurch Überlagerungen entstehen, die wichtige Strömungseigenschaften verdecken können. Somit kann in diesem Fall nicht wie bei Stromlinien durch die Wahl eines geeigneten Verfahrens eine Überdeckung verhindert werden.



(a) **(b)**

Abbildung 1.2: In Bild (a) werden Stromlinien dargestellt. Sie beschreiben die Strömung zu konstanter Zeit. Die Linien überlagern sich nur, wenn sie zu dicht zusammen liegen. In Abbildung (b) werden Bahnlinien dargestellt. Es ist deutlich zu erkennen, dass sich die Linien kreuzen können und somit mehr Überlagerungen möglich sind.

Deswegen befasst sich diese Arbeit mit der Visualisierung von instationären Strömungen um durch Überlagerung verloren gegangene Informationen zurückzugewinnen und damit eine bessere Beschreibung zu erzielen. Dabei beschränkt sie sich auf zweidimensionale zeitabhängige Strömungsfelder und die Visualisierung von Bahn- und Streichlinien.

Durch die Verwendung von Transparenz sind neue Strukturen entstanden, welche mit implementierten Methoden analysiert und bewertet wurden. In dieser Arbeit wurde ein Programm entwickelt, mit dem eine große Anzahl von charakteristischen Linien zu einem beliebigem Vektorfeld erzeugt und analysiert werden kann. Für die Implementierung der Bedienoberfläche wurde Qt verwendet und für die Grafikausgabe OpenGL. Die verwendete Programmiersprache ist C++.

2 Stand der Forschung

2.1 Strömungsvisualisierung

Zur Visualisierung von Strömungen in der Computergrafik existieren viele Techniken. Diese können in vier Kategorien eingeteilt werden. Im Folgenden werden zu jeder dieser Kategorien einige Techniken referenziert mit denen zweidimensionale zeitabhängige Strömungen visualisiert werden können.

2.1.1 Direkte Strömungsvisualisierung

Hierbei werden die Daten ohne große Vorberechnung verarbeitet. Wie beispielsweise bei der Berechnung eines Farbverlaufes um die Strömungsgeschwindigkeiten darzustellen oder durch Pfeile die Strömungsrichtung anzuzeigen.[LH04]

2.1.2 Dichte, Texturbasierte Strömungsvisualisierung

Bei der Texturbasierten Strömungsvisualisierung wird eine Textur verwendet um ein dichtes Strömungsbild darzustellen. Einen Überblick zu den Texturbasierten Strömungsvisualisierungs-Techniken bietet [LHD+04]. Die dort erwähnten Darstellungstechniken die zweidimensionale zeitabhängige Strömungen darstellen können sind:

Unsteady Spot Noise von De Leeuw und Van Lire [LL99]. Hierbei handelt es sich um eine Weiterentwicklung von Spot Noise von Van Wijk [Wij91]. Mit dieser Technik wird eine Textur erzeugt, indem eine Menge von Intensitätsfunktionen an zufälligen Positionen innerhalb der Textur verteilt werden. Dabei wird die Form der Intensitätsfunktionen in Richtung der Strömung verzerrt.

Die folgenden Algorithmen sind eine Weiterentwicklung von der Technik "Line Integral Convolution"(LIC). Hierbei wird eine Textur mit weißem Rauschen verwendet und entlang der berechneten Stromlinien wird die Farbinformationen der Textur anhand der entsprechenden Position gemittelt:

Dynamic LIC (DLIC) von Sunquist [Sun03], Unsteady Flow LIC (UFLIC) von Shen und Kao [SK98], Accelerated UFLIC (AUFLIC) von Lui et al. [LMJI02],

In den folgenden Techniken werden Texel oder Gruppen von Texeln bewegt:

Lagrangian-Eulerian Advection (LEA) von Jobard et. al[JEH01], Die Idee von Max und Becker Texturen zu bewegen[MB96]. Image Based Flow Visualization (IBFV) von Van Wijk [Wij02], Image Space Advection (ISA) Laramee et al. [LJH03], Lagrangian-Eulerian Advection (LEA) von Jobard et al. und Unsteady Flow Advection-Convolution (UFAC) von Weiskopf et al. [WEE03]

Zusätzlich werden die "Texture Transport"Methode von Becker und Rumpf [BR98], in der ein definiertes Muster entlang der berechneten Strömung transportiert wird und die Technik von Bürkle et al. [BPR01] die eine anisotrope Diffusionsmethode erweitert, um instationäre Strömung darzustellen, vorgestellt.

2.1.3 Geometrische Strömungsvisualisierung

Bei der geometrischen Strömungsvisualisierung wird der Verlauf masseloser Partikel in einer gegebenen Strömung berechnet. Sie wird auch integrationsbasierte Strömung genannt. Einen Überblick zu geometrischen Strömungsvisualisierungstechniken geben McLoughlin et al. [MLP+09].

Jobard und Lefer [JL00] präsentieren eine Methode um instationäre Strömungen darzustellen, indem eine Serie von Stromlinienbildern, die die Strömung zu nacheinander liegenden Zeitpunkten beschreibt, erzeugt wird. Durch Verwendung einer Textur, deren Texturkoordinaten sich entlang der Strömung bewegen, wird der Strömungsverlauf visualisiert.

Flächenbasierte Strömungsdarstellung für instationäre Strömung

Bei der flächenbasierten Strömungsvisualisierung werden Flächen berechnet, die den Strömungsverlauf beschreiben. Schafhitzel et al. stellen eine Punkt-basierte Strom- und Bahn-Flächen Konstruktions- und Visualisierungsmethode vor.

Weinkauf und Theisel erzeugen Zeit und Streichflächen in instationären Strömungen [WT12]. Dabei erzeugen sie eine zusammenhängende Fläche, die anhand der Position der masselosen Partikel aufgespannt wird. Die Erzeugung dieser Fläche wird auf der CPU durchgeführt. Diese Technik ist auch geeignet 3D Strömungen darzustellen.

Weitere Techniken, um für instationäre Strömungen Flächen zu berechnen, sind: Point-based stream surfaces and path surfaces [STWE07]

Saattechniken für Stromlinien

Saattechniken sind Methoden, durch die Startpositionen der zu erzeugenden charakteristischen Linien zugewiesen werden. Schlemmer et al. führen einen Ansatz ein, bei dem die Dichte der Stromlinien durch eine skalare Funktion gesteuert wird. [SHH+07] Yu et al. gruppiert Stromlinien, die wichtige Strömungseigenschaften enthalten zu einer Hierarchie, aus der dann Stromlinienbündel zu verschiedenen Detailstufen extrahiert werden. [YWSC12]

2.1.4 Merkmalsbasierte Strömungsvisualisierung

Bei der merkmalsbasierten Strömungsvisualisierung werden lediglich die für den Betrachter interessanten Regionen visualisiert. Dies hat den Vorteil, dass Ressourcen gespart werden können und Informationen, die nicht von Belang sind, entfernt werden können. Eine Kategorisierung verschiedener Techniken bieten Laramee et al.[LHZP07]. Hierbei sind die folgenden Techniken für instationäre Strömungen geeignet: Helman and Hesselink [HH89] stellen eine Technik vor um kritische Punkte zu erkennen und zu klassifizieren. Sadarjoen und Post [SP00] führen zwei Methoden ein um Wirbel aufzuspüren. Dazu werden Stromlinien verwendet. Tricoche et al. identifizieren und stellen topologische Transitionen dar [HTS01].Theisel und Seidel führen eine Tracking Methode, basierend auf Stromlinien ein [TS03]. Theisel et al. prüfen die Bahnlinienorientierte Topologie als neuen Visualisierungsansatz [TWHS04] [TWHS05].

Eine Methode, die sich auch ebenfalls mit der Darstellung von überlagerten, dicht gesäten Linien befasstbefasst, wird im Artikel von Kuhn et al. [KLG+13] beschrieben. Die dort beschriebene Methode ist für die Darstellung dreidimensionaler Strömungen konzipiert und verwendet additives und subtraktives Blending um die sich überlagernden Farbwerte zu mischen.

In dieser Arbeit wird jedoch eine andere Methode speziell für zweidimensionale Vektorfelder entwickelt, bei der es zusätzlich mögliche ist, die Reihenfolge der sich zeitlich überlagernden Linien anhand der klassischen Farbmischung darzustellen, was bei additiven und substractiven Blending nicht möglich ist.

3 Grundlagen

3.1 Bahnlinien und Streichlinien

Bahnlinien werden verwendet um den Verlauf eines masselosen Partikels innerhalb einer gegebenen Strömung zu beschreiben. Die Strömungsdaten, die für diese Arbeit zur Verfügung gestellt wurden, werden durch zweidimensionale zeitabhängige Vektorfelder beschrieben. Diese Vektorfelder enthalten die Geschwindigkeiten der Strömung zu verschiedenen Zeitpunkten. Eine gute Beschreibung um den Weg eines masselosen Partikels in einem Strömungsfeld zu berechnen, geben Laramee et al. [LHD+04]. Hierbei werden die Strömungsdaten definiert durch:

$$\vec{v}(\vec{p}, t) : \Omega \times \Pi \rightarrow \mathbb{R}^n \tag{3.1}$$

dabei entspricht $p \in \Omega \subseteq \mathbb{R}^n$ der Raumposition und $t \in \Pi \subseteq \mathbb{R}$ der Zeitposition innerhalb der Strömungsdaten. Da $v = \frac{d\vec{p}}{dt}$ gilt kann der Weg eines masselosen Partikels definiert werden durch:

$$\vec{p}(t) = \vec{p_0} + \int_0^t \vec{v}(\vec{p}(\tau), \tau)\, \mathrm{d}\tau \tag{3.2}$$

Dabei entspricht $\vec{p}_0$ der Position des Saatpunktes zum Startzeitpunkt 0.

Für die Integration wird in dieser Arbeit das Runge-Kutta-Verfahren angewendet, welches im nächsten Abschnitt beschrieben wird.

Eine Streichlinie ist eine Verbindungslinie zwischen masselosen Partikeln, die zeitlich versetzt an der gleichen Position gesät wurden (siehe Abbildung 3.1).

Der Effekt, der bei der Darstellung von Streichlinien entsteht, entspricht einem Farbfaden, der beim Aussetzen eines Farbkleckses in der Strömung entsteht[SA07]. Um Streichlinien zu erzeugen, kann man beispielsweise die Positionen der masselosen Partikel mit der Startbedingung $\vec{p}_0(t_{00}) = \vec{p}_1(t_{01}) = ... = \vec{p}_{n-1}(t_{0n-1}) = \vec{p}_n(t_{0n})$ wobei $t_{00} \neq t_{01} \neq ... \neq t_{0n-1} \neq t_{0n}$ und sie durch eine Linie miteinander verbinden.

In Abbildung 3.2 wurden Bahn- und Streichlinien in derselben Strömung erzeugt. Dabei sind zwei vollkommen unterschiedliche Strömungsbilder entstanden.

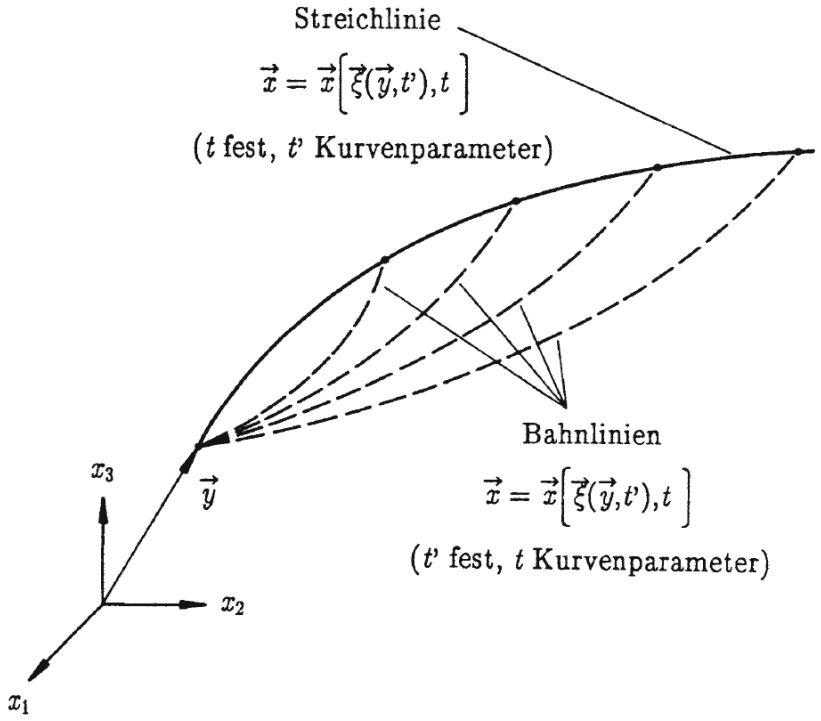


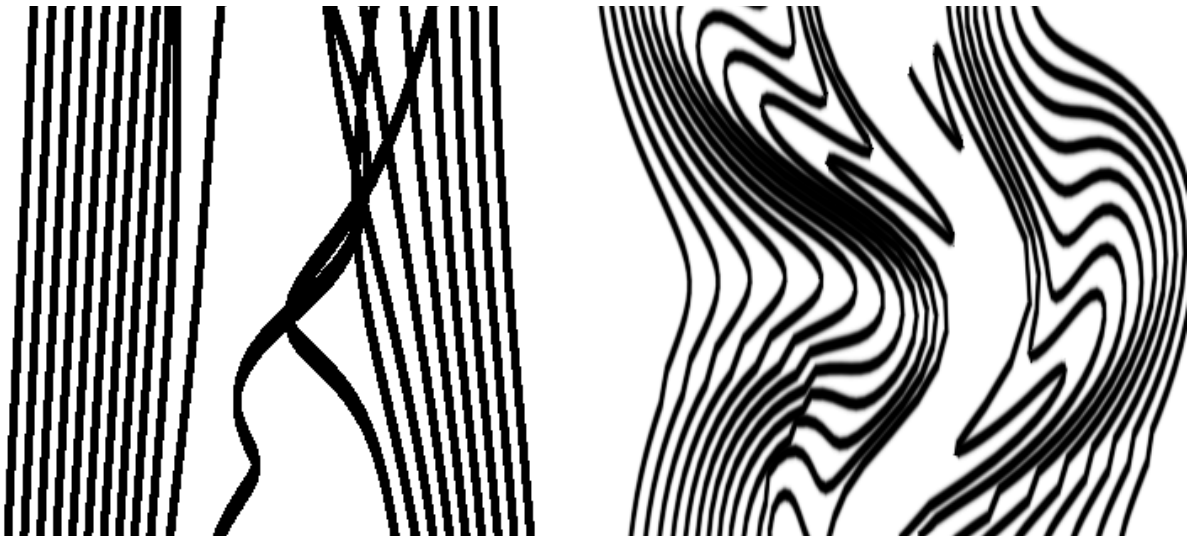
Abbildung 3.1: Streichlinien und Bahnlinien (Quelle: [SA07])

3.2 Runge-Kutta-Verfahren

Das klassische Runge-Kutta-Verfahren (siehe Formel: 3.3) ist ein numerisches Verfahren für die Integration von gewöhnlichen Differenzialgleichungen. Es bezieht im Vergleich zum Eulerschen Polygonzugverfahren mehr Stützstellen für einen Integrationsschritt ein, und erreicht dadurch eine wesentlich höhere Genauigkeit.

$$
\begin{aligned}
k_1 &= f(v_{nx}, v_{ny}, v_{nz}), \\
k_2 &= f(v_{nx} + \frac{1}{2} * h * k_{1x}, v_n y + \frac{1}{2} * h * k_{1y}, tn + \frac{1}{2} * h * k_{1z}), \\
k_3 &= f(v_{nx} + \frac{1}{2} * h * k_{2x}, v_n y + \frac{1}{2} * h * k_{2y}, t_n + \frac{1}{2} * h * k_{2z}), \\
k_4 &= f(v_x + h * k_{3x}, v_{ny} + h * k_{3y}, t_n + h * k_{3z}, t_n + h * k_{3z}) \\
v_{n+1} &= v_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)
\end{aligned}
\tag{3.3}
$$

$k_{1..4}$ stellen die vier Stützstellen dar, die für die Rechnung mit einbezogen werden. Der Fehler dieses Verfahrens liegt bei der Größenordnung $\mathcal{O}(h^5)$. Deshalb verfügt das Verfahren bei der Wahl einer geeigneten Schrittweite über eine sehr hohe Genauigkeit [BM08].



(a) **(b)**

Abbildung 3.2: In Abbildung (a) werden Bahnlinien dargestellt, in Abbildung (b) Streichlinien. Beide Linienbilder beschreiben dieselbe Strömung auf unterschiedliche Weise.

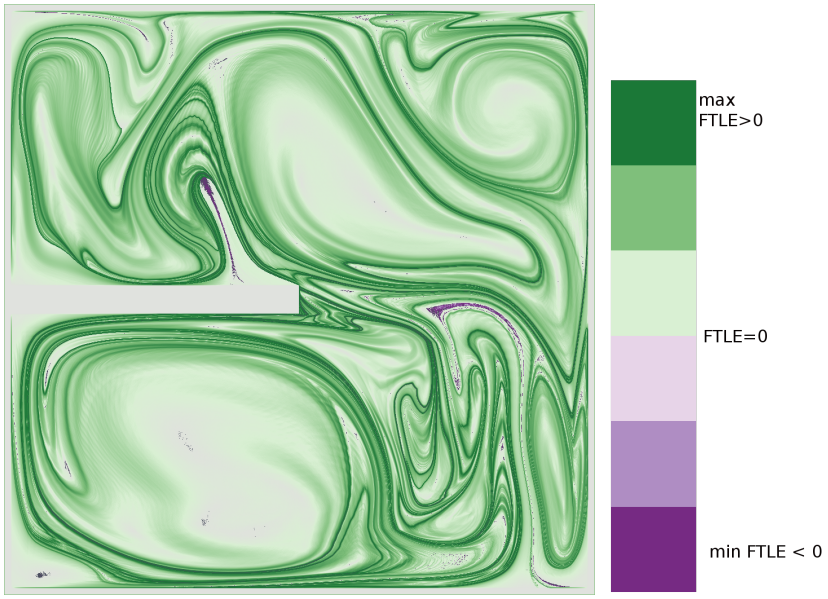
3.3 Finite-Time Lyapunov Exponent

Der Finite-Time Lyapunov Exponent (FTLE) ist ein skalarer Wert, der die Separation benachbarter masseloser Partikel einer gegebenen Strömung nach einer vorgegebenen Zeit beschreibt. Er kann mit der Formel 3.4 berechnet werden. Hierbei entspricht x der Position im Strömungsfeld zum Zeitpunkt t und T der Zeit, in der sich die Partikel in der Strömung frei bewegt haben. $\sqrt{\lambda_{\max}(\Delta)}$ ist die Spektralnorm des rechten Cauchy-Green Deformationstensors Δ und beschreibt die größte Separation der Nachbarpartikel von x. Δ kann durch die Formel 3.5 berechnet werden. Wobei $\frac{d\phi_t^{t+T}}{dx}$ durch die Jacobi-Matrix 3.6 berechnet werden kann.

$$\sigma_t^T(x) = \frac{1}{|T|} \ln \sqrt{\lambda_{\max}(\Delta)} \tag{3.4}$$

hierbei entspricht $\lambda_{\max}(\Delta)$ dem größten Eigenwert des rechten Cauchy-Green Deformationstensors von Δ.

$$\Delta = \frac{d\phi_t^{t+T}}{dx}^\top \frac{d\phi_t^{t+T}}{dx} \tag{3.5}$$



(a) **(b)**

Abbildung 3.3: In diesem FTLE Bild (a) wurden die FTLE Werte von 500*500 Partikeln berechnet. Die Ergebniswerte werden durch den Farbverlauf (b) dargestellt. Dabei entsprechen der kleinste negative FTLE-Wert der Textur-Koordinate 0, der FTLE-Wert 0 der Textur-Koordinate 0,5 und der größte Wert der Koordinate 1.0

$$\left.\frac{d\phi_t^{t+T}}{dx}\right|_{x_i,j} = \begin{bmatrix} \frac{x_{i+1,j}-x_{i-1,j}(t+T)}{x_{i+1,j}(t)-x_{i-1-j}(t)} & \frac{x_{i+1,j}-x_{i-1,j}(t+T)}{x_{i+1,j}(t)-x_{i-1-j}(t)} \\ \frac{x_{i+1,j}-x_{i-1,j}(t+T)}{x_{i+1,j}(t)-x_{i-1-j}(t)} & \frac{x_{i+1,j}-x_{i-1,j}(t+T)}{x_{i+1,j}(t)-x_{i-1-j}(t)} \end{bmatrix} \tag{3.6}$$

Die berechneten FTLE-Werte können dann anhand eines Farbverlaufes zu den jeweiligen Positionen in der Strömung dargestellt werden (siehe Abbildung 3.3).

4 Implementierungsdetails

Der Ansatz für diese Arbeit ist die Erzeugung einer hohen Anzahl von Linien, die sich gegenseitig überlagern, da sie gleichzeitig in einer instationären Strömung erzeugt werden sollen. Hierbei treten die folgenden Probleme auf:

1. Nicht immer kann der gesamte Strömungsverlauf dargestellt werden, da dieser zu komplex ist. Daher sind unterschiedliche Selektionstechniken nötig um bestimmte Bereiche einer Strömung zu analysieren.
2. Um den zeitlichen Verlauf darzustellen, reicht unter Umständen die einfache Darstellung nicht aus. Daher könnte die Verwendung von Farbverläufen hilfreich sein.
3. Die Linien können nicht einfach nacheinander gezeichnet werden, da die Linien gleichzeitig entstehen und sich gegenseitig zu unterschiedlichen Zeiten überlagern und sonst der zeitliche Verlauf nicht durch die Überlagerung der Linien dargestellt werden könnte.
4. Für die Darstellung der Linien ist ein hoher Speicher- und Rechenaufwand nötig, da in dieser Arbeit eine sehr hohe Anzahl von Bahn- und Streichlinien erzeugt wird um den Strömungsverlauf so genau wie möglich zu erfassen.
5. Durch das Zeichnen einer sehr hohen Anzahl von Linien können sich Rasterisierungsprobleme der einzelnen Linien durch Überlagerung verstärken und dadurch das Ergebnisbild verfälschen.

In den folgenden Abschnitten wird auf die Lösung dieser Probleme näher eingegangen.

4.1 Berechnung der Streich- und Bahnlinien

Für die Berechnung der Strömungsbilder ist ein sehr hoher Rechenaufwand nötig, da eine enorme Anzahl von Partikelverläufen berechnet werden muss. Jedoch können die einzelnen Strömungslinien parallel berechnet werden, da sie unabhängig voneinander verlaufen.

Um diesen Vorteil nutzen zu können, werden daher die Berechnungen mit der GPU, also auf der Grafikkarte, durchgeführt, da diese im Gegensatz zur CPU speziell für parallele Berechnungen optimiert ist. In Abbildung 4.1 wird die Anzahl der Gleitkommaoperationen pro Sekunde zweier GPUs und zweier CPUs dargestellt. Es ist deutlich zu erkennen, dass die Anzahl der Operationen pro Sekunde einer GPU weitaus höher ist, als die einer CPU.

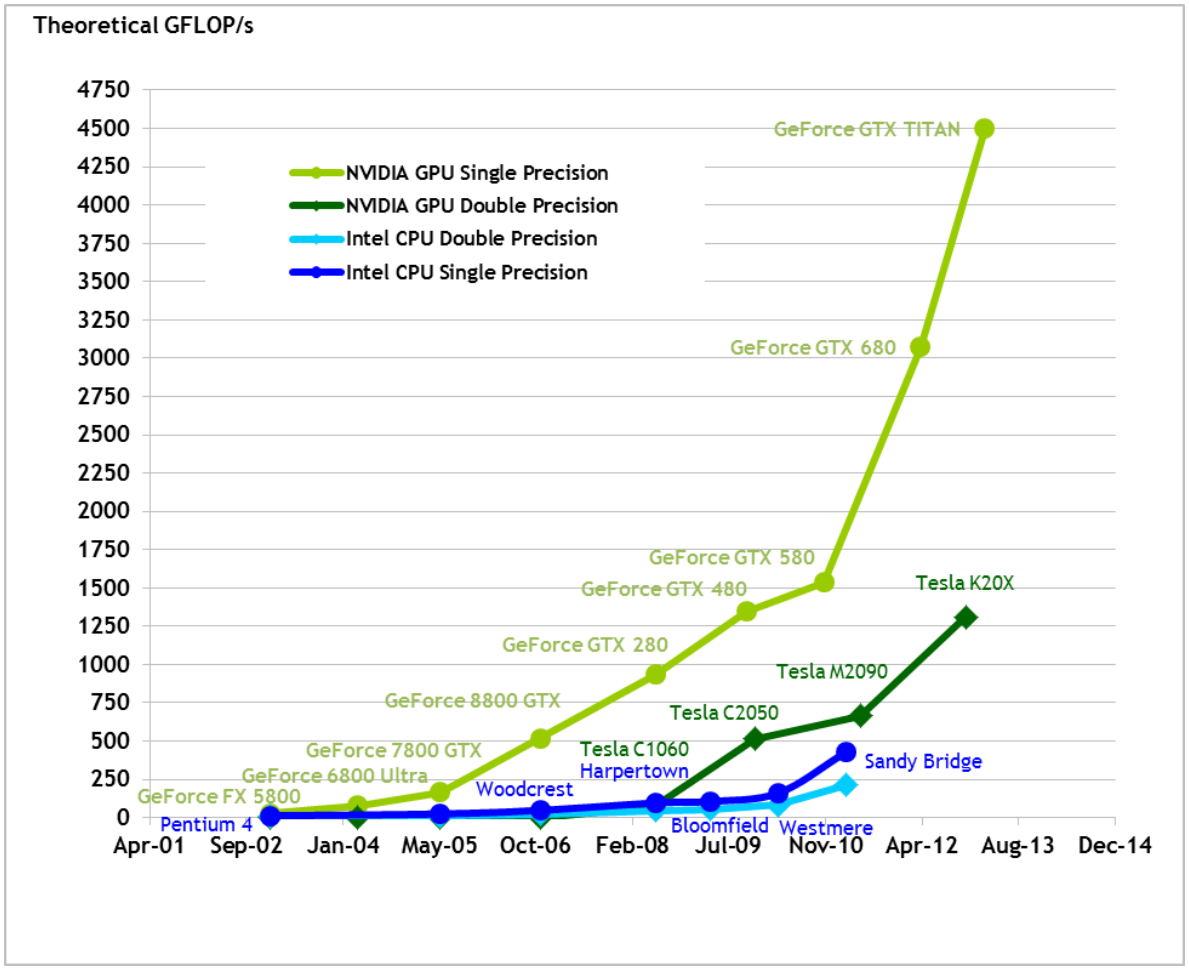


Abbildung 4.1: In dieser Abbildung wird die Entwicklung der Anzahl von Gleitkommaoperationen pro Sekunde zweier GPUs und zweier CPUs im zeitlichen Verlauf dargestellt. Es ist zu erkennen, dass mittlerweile die GPUs deutlich mehr Rechenoperationen für einen festgelegten Zeitbereich durchführen können als die CPUs Quelle: [Cor13]

Zudem ist auch die Datentransferrate auf der GPU deutlich höher als die einer CPU, was in Abbildung 4.2 verdeutlicht wird.

Der Grund für die unterschiedliche Anzahl an Gleitkommaoperationen pro Sekunde ist der, dass die GPU für rechenintensive, hoch parallele Berechnungen spezialisiert ist. Deswegen sind mehr Ressourcen für die Verarbeitung von Daten verfügbar und werden nicht für Daten-Caching oder die Flusskontrolle verwendet [Cor13].

Da zum Erstellen des Resultatbildes eine sehr hohe Anzahl von Linien gezeichnet werden muss, kann das Problem auftreten, dass der Speicherplatz der Grafikkarte nicht ausreicht um alle Vertex-Koordinaten auf einmal zu speichern. Zudem müssen sich die Linien in der

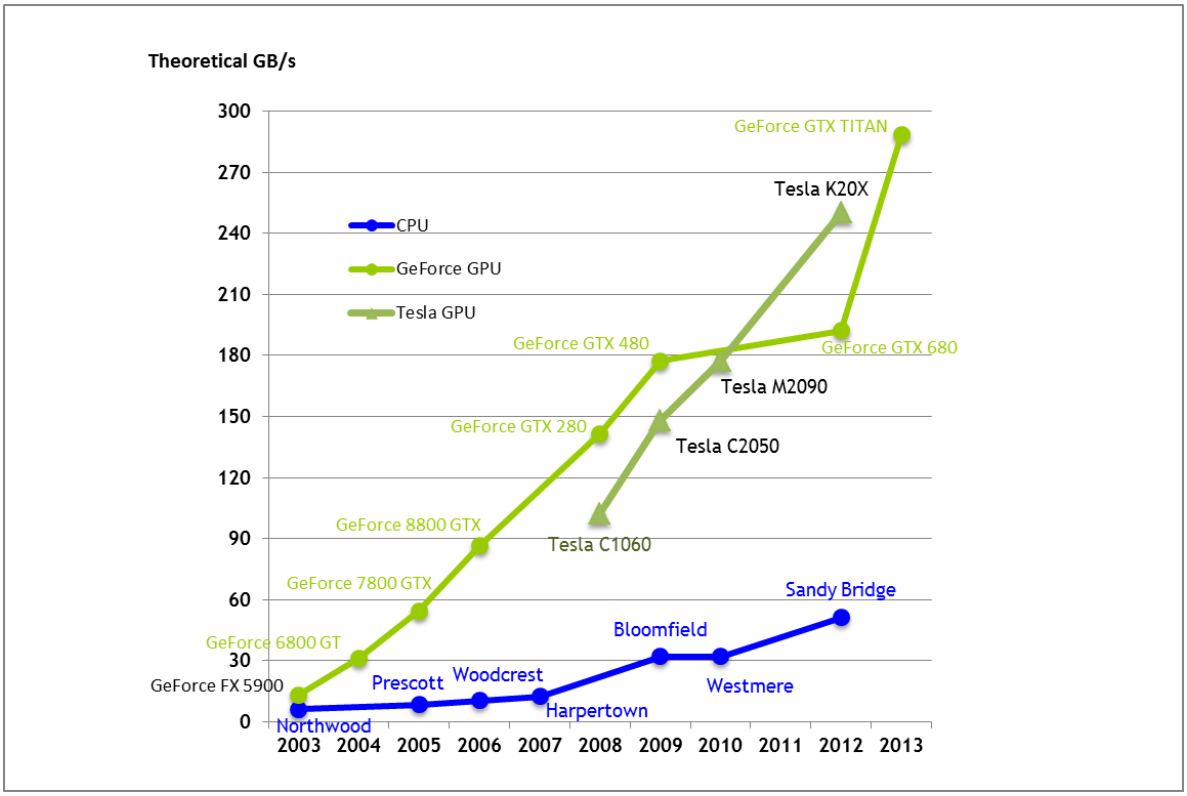


Abbildung 4.2: In diesem Bild wird die Entwicklung der Datentransferrate von GPUs und CPUs dargestellt. Man sieht, dass die Transferrate der GPUs deutlich höher ist als die der CPUs. Quelle: [Cor13]

richtigen Reihenfolge überlagern, da sonst anhand der Überlagerung der zeitliche Verlauf nicht mehr interpretiert werden kann.

Diese Probleme werden durch das folgende Multipass-Verfahren gelöst.

4.1.1 Multipass-Verfahren

Das in dieser Arbeit verwendete Multipass-Verfahren löst das Problem des begrenzten Speicherplatzes auf der Grafikkarte und ermöglicht zudem eine zeitlich korrekte Darstellung. Mit diesem Verfahren wird das Ergebnisbild durch mehrstufiges Zeichnen erzeugt. Das bedeutet, dass die Liniensegmente eines Integrationsschritts berechnet werden und diese dann sofort gezeichnet werden. Somit können die nicht mehr benötigten Vertex-Koordinaten mit den Koordinaten des nächsten Zeitschrittes überschrieben werden.

Die Berechnung der Liniensegmente wird in dieser Arbeit vom Compute-Shader (der im Absatz 4.1.2 näher erläutert wird) durchgeführt. Die berechneten Koordinaten eines Integrationsschritts werden dann direkt an die Grafikpipeline übergeben und bleiben somit auf der Grafikhardware. Die an die Grafikpipeline übergebenen Informationen werden

dann in einen hoch-aufgelösten Framebuffer gezeichnet. Nach dem Zeichnen wird der Speicherplatz für die verwendeten Vertex-Koordinaten wieder dem Compute-Shader zum Berechnen des nächsten Zeitschritts zur Verfügung gestellt. Somit kann der Datenaustausch zwischen CPU und GPU zwischen den Zeichenschritten auf ein Minimum beschränkt werden. Dies führt zu einer enormen Verkürzung der Rechenzeit.

4.1.2 Compute-Shader

Im Gegensatz zu den anderen Shadern ist der Compute-Shader kein Bestandteil der Grafikpipeline. Durch ihn können Berechnungen auf der GPU durchgeführt werden, die nicht zwangsläufig etwas mit der Grafikerzeugung zu tun haben. Der Compute-Shader hat keine festgelegten Inputs oder Outputs. Mit ihm kann daher direkt auf Speicher und Texturen auf der Grafikkarte zugegriffen werden [SSKLK13].

4.2 Saattechniken

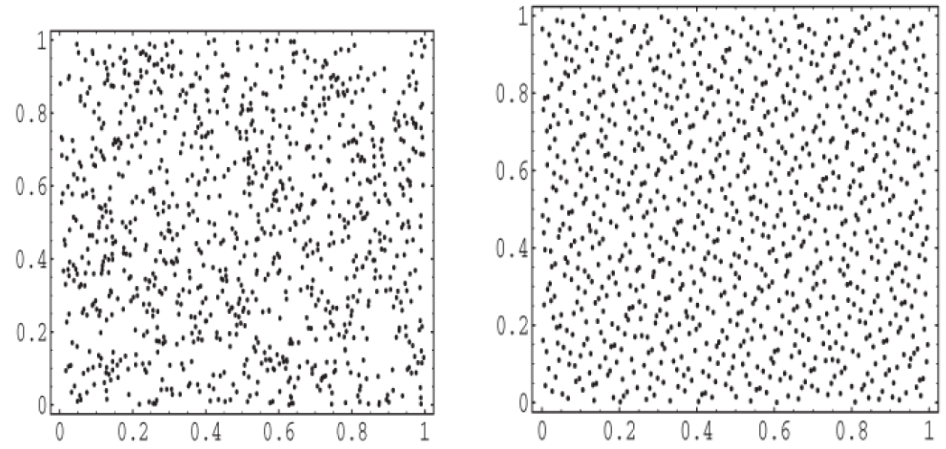
Die in dieser Arbeit implementierten Saattechniken unterstützen das Setzen von beliebig vielen Saatpunkten auf einer Linie oder in einem Feld. Die Anordnung dieser Punkte kann dabei zufällig oder gleichmäßig sein. Die Saattechniken können dabei beliebig kombiniert werden.

4.2.1 Erzeugung zufälliger Saatpunkte

Um die Position der Saatpunkte innerhalb des selektierten Bereichs zufällig zu setzen, werden Pseudozufallszahlen zwischen 0 und 1 verwendet. (Dabei muss beachtet werden, dass die Zufallszahlen so gleichmäßig wie möglich verteilt sind, da ansonsten Artefakte durch die ungleichmäßige Verteilung der Saatpunkte im Resultatbild entstehen können.) Die erzeugte Zufallszahl kann dann verwendet werden um die Position des Saatpunktes zu bestimmen. Zusätzlich werden Saatpunkte anhand der Halten-Folgen verteilt, um eine höhere Gleichverteilung beim Säen zu erhalten.

4.2.2 Van der Corput- und Halton-Folgen

Die Van der Corput-Folge und die Halton-Folgen sind Folgen, die eine kleine Diskrepanz aufweisen. Um das n-te Folgeglied x_n der Van der Corput-Folge zur Basis b zu bestimmen,



(a) **(b)**

Abbildung 4.3: In Abbildung (a) wird die Verteilung von Pseudozufallszahlen dargestellt, in Abbildung (b) die der Halton-Folge mit Basen 2 und 3 (Quelle: [HA09])

kann man die Formel 4.1 verwenden. Dabei ist a_j die jeweilige Ziffer der Zahl n zur Basis b also $n = \sum_{j=0}^{\infty} a_j(n)b^j$ und $x_n = \phi_b(n) \forall n \geq 0$ [HA09]

$$\phi_b(n) = \sum_{j=0}^{\infty} a_j(n)b^{-j-1}$$
$$Quelle : [\text{HA09}] \tag{4.1}$$

Die Halton-Folge erhält man, wenn man die Konstruktion der Van der Corput-Folge auf $s > 1$ Dimensionen erweitert (siehe Formel 4.2). Dabei müssen für b_s unterschiedliche Primzahlen als Basis für die Konstruktion verwendet werden.

$$x_n = (\phi_{b_1}(n), ..., \phi_{b_s}(n)) \forall n \geq 0$$
$$Quelle : [\text{HA09}] \tag{4.2}$$

In Abbildung 4.3 wird die Verteilung der ersten 1000 Punkte zweidimensionaler Folgen mit Pseudozufallszahlen und Halton-Folge dargestellt. Die Halton-Folge füllt den Bereich viel gleichmäßiger aus als die Pseudozufalls-Folge [HA09].

4.3 Farbverläufe

Durch das Verwenden von Farben können zusätzliche Informationen dargestellt werden. Beispielsweise ist es möglich, den zeitlichen Verlauf der Strömung durch die Wahl eines geeigneten Farbverlaufs darzustellen. Jedoch kann die Wahl eines ungeeigneten Farbverlaufs zu Fehlinterpretationen führen, da beispielsweise durch Kontrastunterschiede Stufen im Farbverlauf entstehen können [DB07]. Deswegen eignen sich lediglich sorgfältig ausgewählte

(a) (b) (c)

Abbildung 4.4: Abbildung (a) zeigt ein divergentes Farbschema. Es eignet sich, um unterschiedliche Extreme darzustellen. Das qualitative Farbschema (b) eignet sich um bestimmte Bereiche optisch voneinander zu trennen. Das lineare Farbschema (c) eignet sich um einen Anstieg von Werten darzustellen. Quelle: [Cyn14]

Farbverläufe, um einen Sachverhalt zu beschreiben. Diese können in drei Kategorien eingeteilt werden: das divergente Farbschema (zur Darstellung unterschiedlicher Extrema) (siehe Abbildung 4.4a), das qualitative Farbschema (zur optischen Trennung von Bereichen) (siehe Abbildung 4.4b) und das lineare Farbschema (zur Darstellung von linearen Verläufen) (siehe Abbildung 4.4c) [Cyn14].

Deswegen kann jeweils ein passender Farbverlauf durch das für diese Arbeit entwickelte Programm erstellt und geladen werden.

4.4 Blending

Blending beschreibt in der Computergrafik die Technik, Farbwerte zu mischen. Dabei werden die folgenden Möglichkeiten von OpenGL unterstützt: Addieren, Subtrahieren, umgekehrte Subtraktion, Minimal-Wert und Maximal-Wert. In dieser Arbeit wird die Standard-Verknüpfungsfunktion GL_FUNC_ADD, also die Addition der Farbwerte, verwendet. Wird zusätzlich zu den Farbwerten RGB ein Alpha-Kanal verwendet, spricht man von Alpha Blending. Der Alpha-Kanal wird verwendet um Transparenz darzustellen.

Transparenz eignet sich nicht nur um durchsichtige Objekte darzustellen, sondern auch um glatte Kanten zu erzeugen. Dies spielt beim Anti-Aliasing (siehe Abschnitt 4.5) eine sehr wichtige Rolle. Im Folgenden wird die Erzeugung von Transparenz in OpenGL erläutert.

Nachdem das zu zeichnende Objekt auf der Grafikkarte rasterisiert und dadurch in Fragmente konvertiert wurde, wird die Farbe des Fragments (R_S, G_S, B_S, A_S) mit der Farbe des im Bildspeicher vorhandenen Pixels (R_D, G_D, B_D, A_D) gemischt [NFHS11]. Durch die Art und Weise wie die Farbinformationen gemischt werden, kann Transparenz realisiert werden. Im Folgenden werden Mischverfahren beschrieben, die im implementierten Programm verwendet werden.

4.4.1 Klassisches Mischen von Farben

Die in Formel 4.3 dargestellte Farbmischung wird am häufigsten verwendet, da mit ihr sehr realistische Bilder von transparenten Flächen vor einem undurchsichtigen Hintergrund gemacht werden können.

$$\begin{aligned} R_m &= A_S * R_S + (1 - A_S) * R_d \\ G_m &= A_S * G_S + (1 - A_S) * G_d \\ B_m &= A_S * B_S + (1 - A_S) * B_d \\ A_m &= A_S * A_S + (1 - A_S) * S_d \end{aligned} \tag{4.3}$$

[NFHS11]

Möchte man eine Farbmischung von transparenten Flächen erzeugen, eignet sich die Porter Duff Composition (Folrmel 4.4) um die resultierende Farbe zu bestimmen [PD84]. Hierbei muss jedoch die Ergebnisfarbe durch den neu berechneten Alpha-Wert geteilt werden, was zusätzliche Rechenzeit kostet.

$$\begin{aligned} R_m &= (A_S * R_S + (1 - A_S) * R_d) * \frac{1}{A_m} \\ G_m &= (A_S * G_S + (1 - A_S) * G_d * \frac{1}{A_m} \\ B_m &= (B_S * R_S + (1 - A_S) * B_d * \frac{1}{A_m} \\ A_m &= A_S + (1 - A_S) * A_d \end{aligned} \tag{4.4}$$

4.4.2 Additives Blending

Beim additiven Blending werden die aufeinander liegenden Farbverläufe addiert. Diese Methode wird im Programm verwendet um zu analysieren, wie oft die berechneten Bahnlinien durch einen Pixel verlaufen (siehe Kapitel: 5.7).

$$\begin{aligned} R_m &= R_S + R_d \\ G_m &= G_S + G_d \\ B_m &= B_S + B_d \\ A_m &= A_S + A_d \end{aligned} \tag{4.5}$$

4.4.3 Additives Blending mit Alpha

Beim additiven Blending mit Alpha wird der Farbwert des Fragments mit seinem Alpha Wert multipliziert und der dem Farbwert des Pixels addiert. Somit erscheinen die Bereiche, die überlagert werden, heller.

$$\begin{aligned} R_m &= R_S * A_S + R_d \\ G_m &= G_S * A_S + G_d \\ B_m &= B_S * A_S + B_d \\ A_m &= A_S * A_S + A_d \end{aligned} \tag{4.6}$$

4.5 Aliasing Problematik

In der digitalen Bildverarbeitung wird ein Bild durch Pixel dargestellt, die in einem Raster angeordnet sind. Um ein Bild zu erzeugen und den Pixeln die richtigen Farbwerte zuzuweisen, muss deswegen die berechnete Geometrie auf das Raster übertragen werden.

4.5.1 Moiré-Muster

Da das berechnete Bild durch eine endliche Zahl von Pixeln dargestellt wird, können Fehler durch Unterabtastung entstehen. Die Pixelanzahl reicht nicht mehr aus, um das Bild vollständig zu beschreiben. Nach dem Abtasttheorem muss die Abtastfrequenz mindestens doppelt so hoch sein wie die höchste vorkommende Ortsfrequenz, um das berechnete Bild ohne Informationsverlust zu rekonstruieren. In Abbildung 4.6a ist zu erkennen, dass durch zu eng zusammen liegende Linien eine zu hohe Frequenz entstehen kann, wodurch Aliasing-Artefakte (sogenannte "Moiré-Muster") entstehen [NFHS11].

4.5.2 Treppenstufen-Effekt

Ein zusätzlicher Effekt, der durch die Rasterisierung entsteht, ist der sogenannte "Treppenstufen-Effekt". Er tritt beispielsweise bei schrägen Linien und Kanten auf und entsteht dadurch, dass die Kanten und Linien nicht glatt, sondern dem Raster entsprechend, stufenartig verlaufen. In Abbildung 4.6a wird der Treppenstufen-Effekt dargestellt. Durch die regelmäßig auftretenden Stufen entstehen ungewollte Muster, die durch Kantenglättung verringert werden können [NFHS11].

4.6 Anti-Aliasing Methoden

Um dem Aliasing-Effekt entgegenzuwirken, gibt es zwei Möglichkeiten. Entweder kann durch eine "Pre-Filterungs-Methode" die höchste Ortsfrequenz entfernt, oder durch "Post-Filterungs-Methoden" das Bild höher abgetastet werden. In dieser Arbeit wurde jeweils zu den beiden Möglichkeiten, die nachfolgend beschrieben werden, ein Verfahren implementiert und angewendet [NFHS11].

4.6.1 Supersampling

Supersampling ist eine "Post-Filterungs-Methode" bei der durch Erhöhung der Auflösung des Bildes die Abtastung des berechneten Bildes erhöht wird. Danach muss durch eine geeignete Methode das Bild wieder verkleinert werden, um die gewünschte Auflösung zu erhalten. Bei der implementierten Methode wurde die Auflösung zur Erzeugung des Bildes verdoppelt. Danach wurde das Bild im Compute-Shader gefiltert, um die Auflösung wieder zu halbieren. Es gibt verschiedene Filterungs-Methoden, die gebräuchlich sind um das Ergebnisbild zu erzeugen. In diesem Fall wurde zur Filterung der Mittelwert der umliegenden 4 Pixel verwendet. Dabei muss sichergestellt werden, dass die für das Bild erzeugten Geometrien an die erhöhte Auflösung angepasst wird, da beispielsweise bei OpenGL die Linienbreite durch die Anzahl an Pixeln festlegt wird. Eine Linie wirkt bei doppelter Auflösung des Bildes somit schmaler, wenn die Anzahl an Pixeln, die die Breite der Linie darstellen, nicht auch verdoppelt wird.

4.6.2 Kanten glätten

Diese Pre-Filterungsmethode wird von OpenGL für alle Grafik-Primitive (Linien, Punkte, Polygone) angeboten. Für sie wird für jedes Fragment der zu zeichnenden Linie ein Überdeckungswert berechnet. Anhand dieses Werts wird dann ein alpha-Wert bestimmt, der dem Pixel zugewiesen wird.

Jedoch gibt es Einschränkungen bei dieser Methode. Beim Zeichnen von "dicken" Linien überlagern sich die Linienfragmente oft teilweise, außerdem können die Linien ohnehin nicht beliebig dick gemacht werden (Siehe Abbildung 4.7). Verwendet man stattdessen Vierecke tritt dieses Problem nicht auf. Dafür beschreiben Nischwitz et al. [NFHS11], dass durch die Kantenglättung bei Vierecken ein Streifen von einer Kante zur anderen entstehen kann, da die Vierecke intern in einen Satz von verbundenen Dreiecken zerlegt werden (siehe Abbildung 4.5). Um dieses Problem zu lösen, wird darauf hingewiesen, die Mischfunktion glBlendFunc (GL_SRC_ALPHA_SATURATE,GL_ONE) zu verwenden und die Liniensegmente von vorne nach hinten zu zeichnen, also die Strömung zeitlich rückwärts zu berechnen.

Bei der Implementierung ist jedoch das Problem aufgetreten, dass der Standardbildspeicher einen zu kleinen Alpha-Kanal aufweist und somit kein ausreichend kleiner alpha-Wert erzeugt werden konnte. Dies hat zur Folge, dass sich keine hohe Anzahl von transparenten Linien überlagern kann, ohne sich vollständig zu überdecken. Bei der Verwendung eines

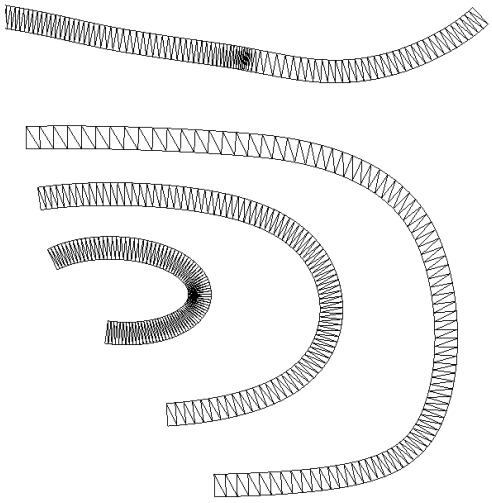
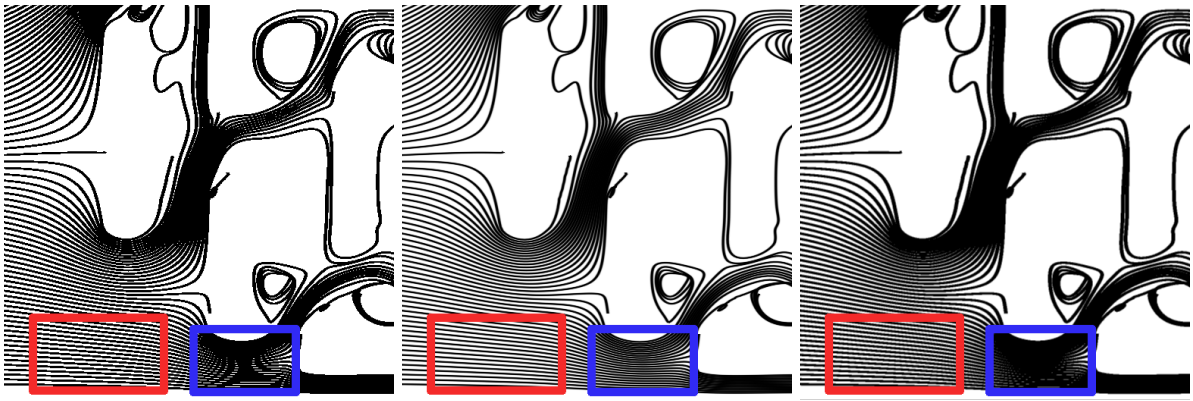


Abbildung 4.5: interne Darstellung der Linien



(a) **(b)** **(c)**

Abbildung 4.6: In Abbildung (a) sind im linken Kasten Artefakte durch den Treppenstufen-Effekt sichtbar und im rechten Kasten Artefakte durch den Moiré-Effekt. In Abbildung (b) wird Antialiasing durch eine Prefiltermethode angewendet. In Abbildung (c) wird Antialiasing durch eine Post-Filterungs-Methode angewendet.

Framebuffers mit ausreichender Größe des Alpha-Kanals funktioniert hingegen die OpenGL Kantenglättung nicht mehr.

Um dieses Problem zu umgehen, wird die Kantenglättung in dieser Arbeit durch eine Texturierung ersetzt, die Informationen über den Transparenzverlauf der Linie enthält. Dabei tritt auch nicht das Problem mit der diagonalen Kante beim Zeichnen von Vierecken auf, und somit muss auch die Methode, die von Nischwitz et al. [NFHS11] beschrieben wurde, nicht angewendet werden.

Zeichnen der Linien

In dieser Arbeit wurden zwei unterschiedliche Arten um Linien zu zeichnen implementiert. Zum einen das einfache Linien zeichnen, bei dem nur Start- und Endposition zum Zeichnen verwendet wird und zum Anderen das Zeichnen von Vierecke. Bei letzterer Variante müssen jeweils die Verbindungskanten zur Start- und Endposition im Geometry-Shader berechnet werden, damit eine Linie gleichmäßiger Dicke entsteht.

4.7 Rasterisierungsproblematik bei überlagerten Linien

4.7.1 Wahl der Linien

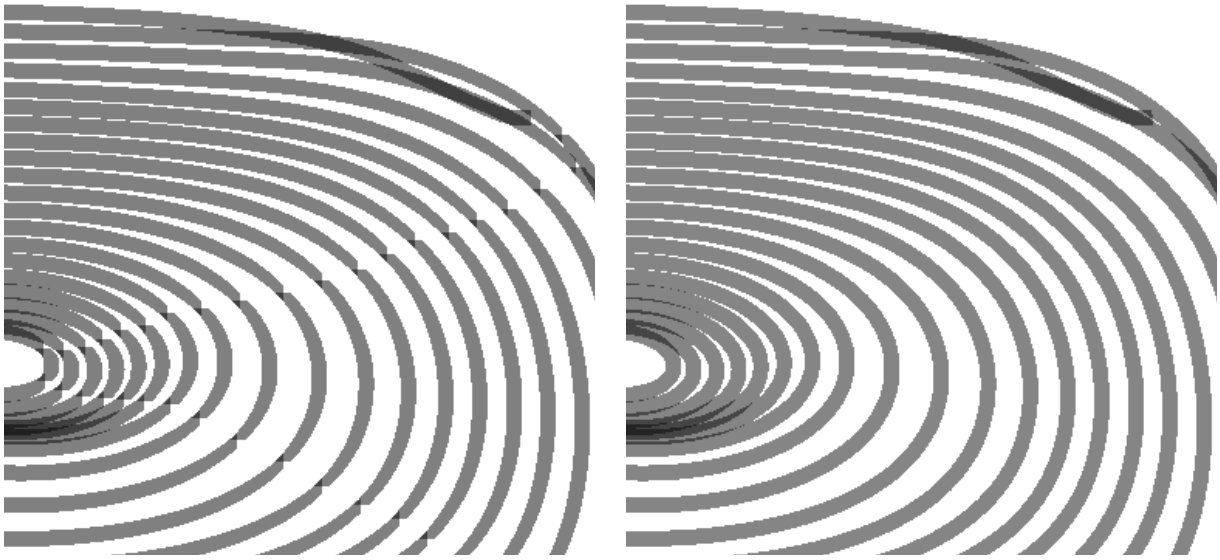
Durch das Erzeugen von transparenten Linien durch einfache Linien in OpenGL, können Strukturen entstehen, die durch Drehen des Sichtfeldes als Rasterisierungsartefakte identifiziert werden können (siehe Abbildung 4.8). Die Problematik ist bereits bei einzelnen Linien zu erkennen, die dick gezeichnet werden. Sobald die Liniensegmente in einem gewissen Winkel zueinander stehen, entstehen Rasterisierungsartefakte die beim Zeichnen von Vierecken nicht entstehen (siehe Abbildung 4.7). Jedoch ist dies nicht die Ursache für das Rasterisierungsartefakt in Abbildung 4.8. In Abbildung 4.9a wird gezeigt, dass der Linienverlauf Auswirkung auf die Liniendicke hat. Dieses Problem, das durch die Rasterisierung von Linien entsteht, kann behoben werden indem keine Linien, sondern Vierecke gezeichnet werden und die Nahtstellen durch die jeweiligen Winkelhalbierenden der beiden Kanten erzeugt werden. Zusätzlich kann ein weicher Linienverlauf generiert werden, wodurch weitere Rasterisierungsartefakte verringert werden. Dies ist in Abbildung 4.10 dargestellt.

Berechnung der Winkelhalbierenden

Um die Winkelhalbierende $\vec{w}_h$ zwischen zwei Liniensegmenten zu berechnen, werden die Vektoren $\vec{v}_0, \vec{v}_1, \vec{v}_2$ benötigt, welche die Liniensegmente aufspannen. Zu den Liniensegmenten wird dann jeweils der orthogonale Vektor $\vec{v}_{ortho_0}$ und $\vec{v}_{ortho_1}$ berechnet. Dabei muss beachtet werden, dass die Vektoren vom Liniensegment betrachtet in dieselbe Richtung zeigen. Durch Normierung und Addition der beiden orthogonalen Vektoren erhält man den Vektor, der den Winkel zwischen den beiden Liniensegmenten teilt (siehe Formel 4.7).

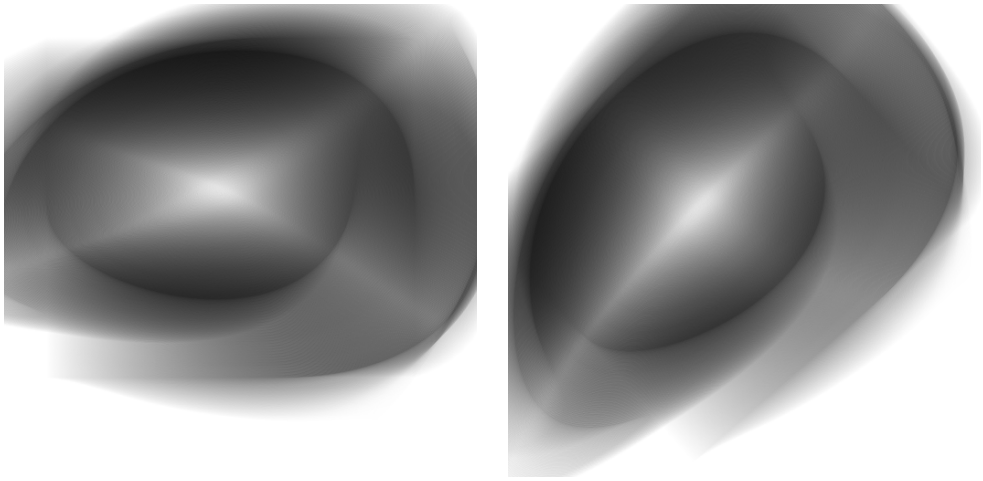
$$\vec{v}_{ortho_0} = \begin{pmatrix} (\vec{v}_1 - \vec{v}_0).y \\ -(\vec{v}_1 - \vec{v}_0).x \end{pmatrix} \quad \vec{v}_{ortho_1} = \begin{pmatrix} (\vec{v}_2 - \vec{v}_1).y \\ -(\vec{v}_2 - \vec{v}_1).x \end{pmatrix} \tag{4.7}$$

$$\vec{w}_h = normal(\vec{v}_{ortho_0}) + normal(\vec{v}_{ortho_1})$$



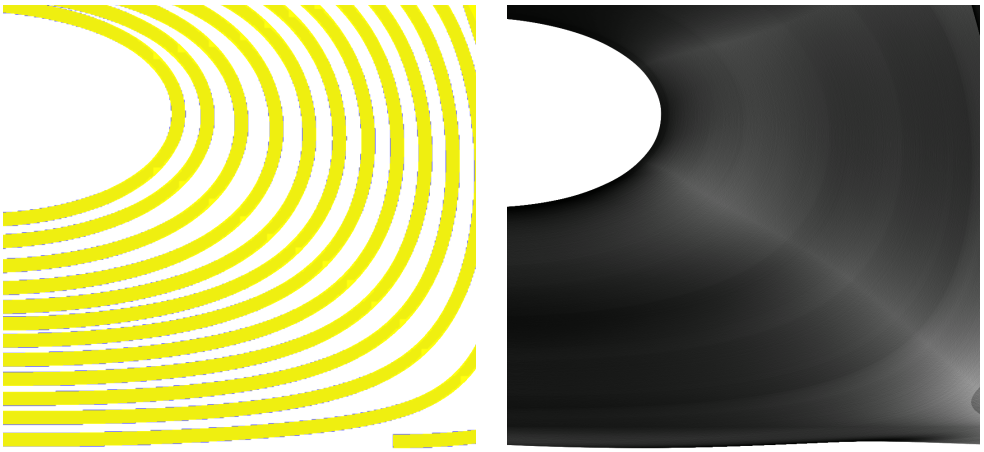
(a) **(b)**

Abbildung 4.7: In Abbildung (a) wurden Linien verwendet um das Bild zu erzeugen. Bei Linien die dicker als ein Pixel sind entstehen Rasterisierungsfehler sobald die Linien in einem gewissen Winkel zueinander stehen. In Abbildung (b) wurden Vierecke verwendet. Hierbei entstehen keine Rasterisierungsfehler.



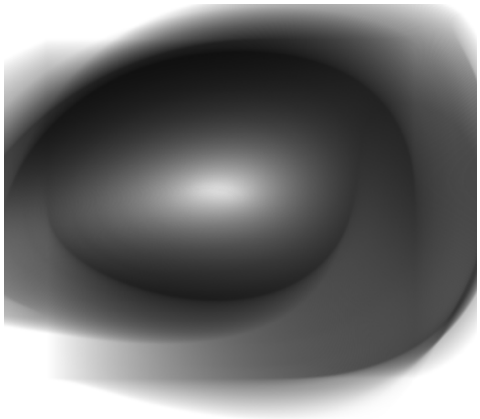
(a) **(b)**

Abbildung 4.8: In Abbildung (a) ist ein weißes Kreuz im Strömungsbild erkennbar. Durch Drehen des Sichtfeldes und wiederholtes Zeichnen erscheint das Kreuz an einer anderen Stelle im Bild. Dies beweist, dass es sich hierbei um ein Rasterisierungsartefakt handelt. (siehe Abbildung (b)).



(a) **(b)**

Abbildung 4.9: In Abbildung (a) wurde die gelben Linien durch Vierecke erzeugt und über die blauen Linien gelegt, die durch Linien erzeugt wurden. Dies zeigt, dass die helle Struktur in (b) durch die Verwendung einfacher Linien entstanden sein muss, da die Linien in diesem Bereich durch weniger Pixel dargestellt werden und somit dieser Bereich eine höhere Transparenz aufweist.



(a) Linienbreite 4 Pixel

Abbildung 4.10: In den beiden Bildern (a) wurden Vierecke verwendet um das Bild zu erzeugen. Dabei wurden die Winkelhalbierenden verwendet um den Übergang zwischen den einzelnen Schritten zu zeichnen.

4.8 Gewichtetes-Saatpunkt setzen

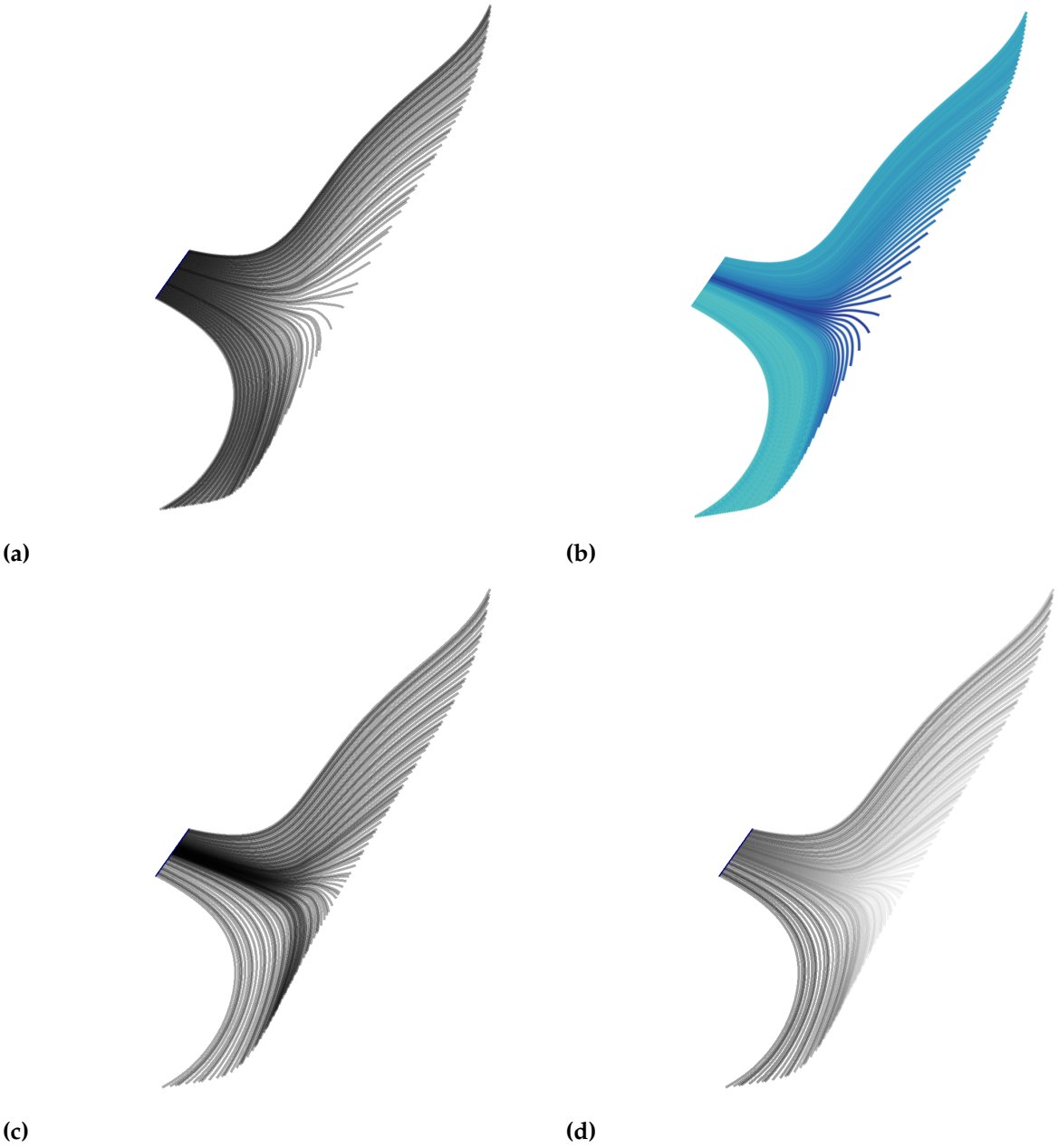
Bei divergenten Strömungen kommt es beim gleichmäßigem Saatpunkt setzen unweigerlich zu dem Effekt, dass in dem divergenten Bereich weniger Bahnlinien enthalten sind (siehe Abbildung 4.11a). Somit müssen mehr Linien gesät werden um Strukturen in diesem Bereich sichtbar zu machen. Umgekehrt kommen auch konvergente Strömungen vor, in denen dann weniger Linien notwendig sind um den Strömungsverlauf zu beschreiben.

Wie bereits in Kapitel 3 erläutert wurde, wird durch den FTLE-Wert die maximale Separation von masselosen Partikeln in einer Strömung beschrieben. Somit kann mit Hilfe des FTLE-Werts das Säen angepasst werden. Die Verteilung der Saatpunkte wird dadurch entsprechend der Konvergenz oder Divergenz der Strömung festgelegt. In Abbildung 4.11b wird anhand eines Farbverlaufs der FTLE-Wert der Startpositionen dargestellt. Durch den tieferen Blauton wird der höhere FTLE-Wert in diesem Bereich erkennbar.

Um dieses Verfahren zu entwickeln, wurde der "Important Sampling" Ansatz verwendet. Das bedeutet, dass die Saatpunkte nicht mehr mit der gleichen Wahrscheinlichkeit im selektierten Bereich verteilt werden, sondern anhand des FTLE Wertes. Dafür muss zunächst der FTLE Wert berechnet werden, wobei hier kein exponentieller Wert verwendet wird, also bei der Berechnung siehe Formel 3.4 kein natürlicher Logarithmus gezogen werden muss. Danach wird ein Speicher mit Verweisen auf die Saatpunktpositionen erzeugt. Je höher die Wahrscheinlichkeit ist, umso mehr Verweise auf den Bereich um den Saatpunkt herum existieren im Speicher. Dabei muss beachtet werden, dass die Verweise nicht auf die selbe Positionen zeigen sondern gleichmäßig verteilt sind. Danach können die Saatpunkt Positionen zufällig aus dem Speicher gezogen werden. In dieser Arbeit wurde als Speicher eine Textur verwendet. Dies hat den Vorteil, dass zwischen den gespeicherten Saatpunkt Positionen interpoliert werden kann und somit mehr unterschiedliche Saatpunkt Positionen erzeugt werden können. Das Ergebnis ist in Abbildung 4.11c zu sehen, es verlaufen deutlich mehr Bahnlinien in dem Bereich in dem sich die Strömung trennt.

Jedoch zeigt Abbildung 4.11c auch, dass durch das ungleichmäßige Säen Transparenzunterschiede entstehen. In dem Bereich, in dem mehr Saatpunkte gesetzt wurden, ist die Transparenz geringer bis die Strömung divergiert und in dem Bereich, in den weniger Linien gesät wurden, ist die Transparenz höher.

Durch Anpassung der Transparenz anhand des erzeugten Gewichts kann dieser Effekt abgefangen werden.



(a) **(b)**

(c) **(d)**

Abbildung 4.11: In Abbildung (a) wurden die Saatpunkte gleichmäßig gesetzt. In dem Bereich, in dem die Strömung divergiert, sind deutlich weniger Linien enthalten. In Abbildung (b) wird der FTLE-Wert an der Startposition anhand eines Farbverlaufes dargestellt. In dem Bereich, in dem die Strömung divergiert, ist der FTLE-Wert höher. In Abbildung (c) wurden anhand des FTLE-Wertes die Saatpunkte gesetzt. Es sind mehr Linien in dem Bereich, in dem die Strömung divergiert, zu erkennen jedoch auch starke Transparenzunterschiede. Aus diesem Grund wurde in Abbildung (d) wurde der alpha-Wert durch den FTLE-Wert angepasst.

5 Analyse-Methoden

Im folgenden Abschnitt werden die unterschiedlichen Techniken beschrieben, mit denen Informationen eines gegebenen Strömungsfeldes analysiert und visualisiert wurden.

5.1 Animation

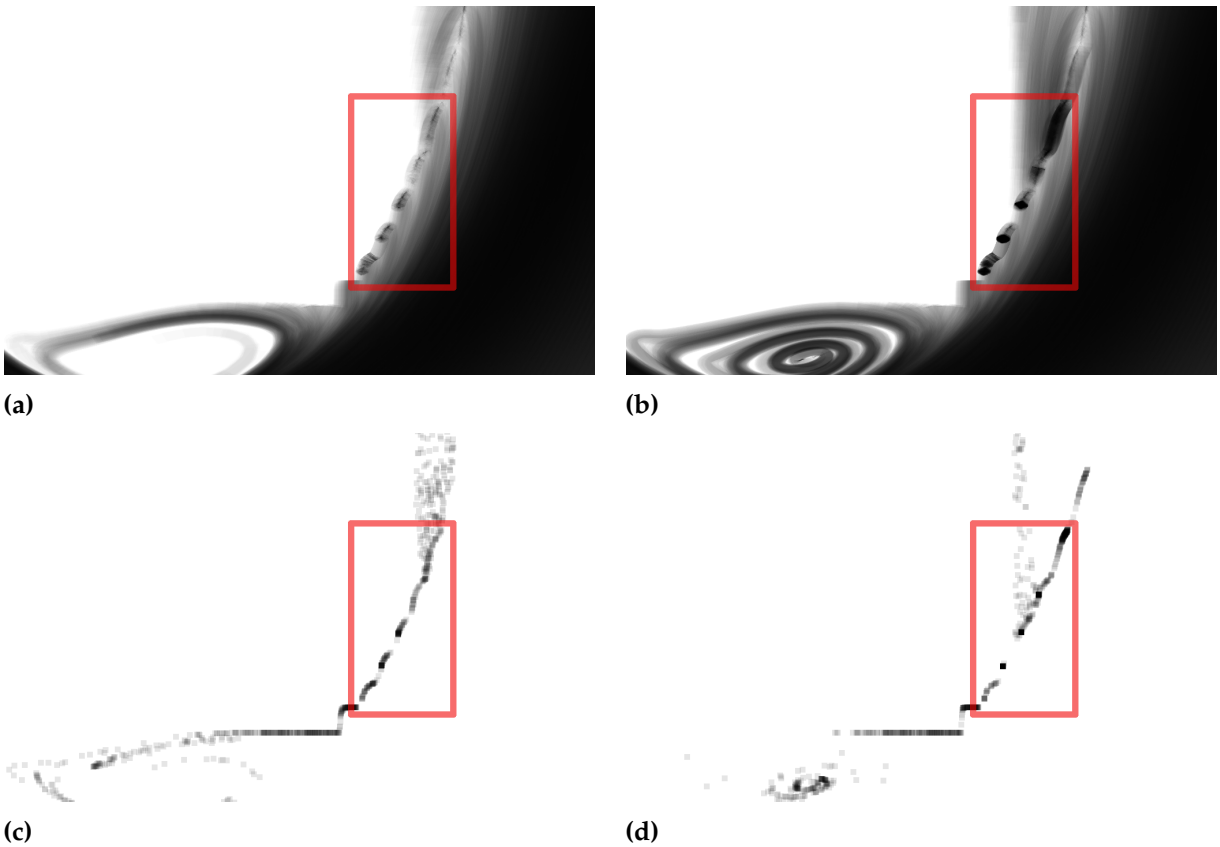
Mithilfe des in dieser Arbeit entwickelten Programms lassen sich durch automatisierte Parameteränderungen wie beispielsweise Änderung von Zeit, Transparenz, Seeding-Dichte usw. rasch neue Strömungsbilder erzeugen. Dies vereinfacht die Suche nach interessanten Strukturen. Außerdem lassen sich so Animationen erzeugen, wodurch die Entstehung von Strukturen ebenfalls besser nachvollzogen werden kann.

5.2 Darstellung von masselosen Partikeln

Um die Entstehung von Strukturen in einem Strömungsbild nachvollziehen zu können, kann es hilfreich sein, anstelle von Linien den animierten Verlauf halbtransparenter masseloser Partikel zu betrachten. In Abbildung 5.1 werden dazu Bahnlinien dargestellt, an denen sich einzelne Bereiche hervorheben. Durch die Darstellung transparenter masseloser Partikel kann veranschaulicht werden, dass sich in diesem Bereich die Partikel sammeln und die Strukturen nicht durch Selbstüberlagerung einer Bahnlinie entstehen.

5.3 Saat-Techniken

Durch die Anwendung verschiedener Saattechniken wird untersucht, welchen Einfluss die unterschiedliche Formen des Saatbereiches auf das Ergebnis haben. Untersucht werden die Saatpunkte in einem Rechteck und einer Linie. Zudem wurden die Saatpunkte gleichmäßig oder zufällig in den ausgewählten Bereichen verteilt. In Abbildung 5.2 werden verschiedene Saat-Möglichkeiten dargestellt. Außerdem wurde ein gewichtetes Säen implementiert (siehe Kapitel 4.8).



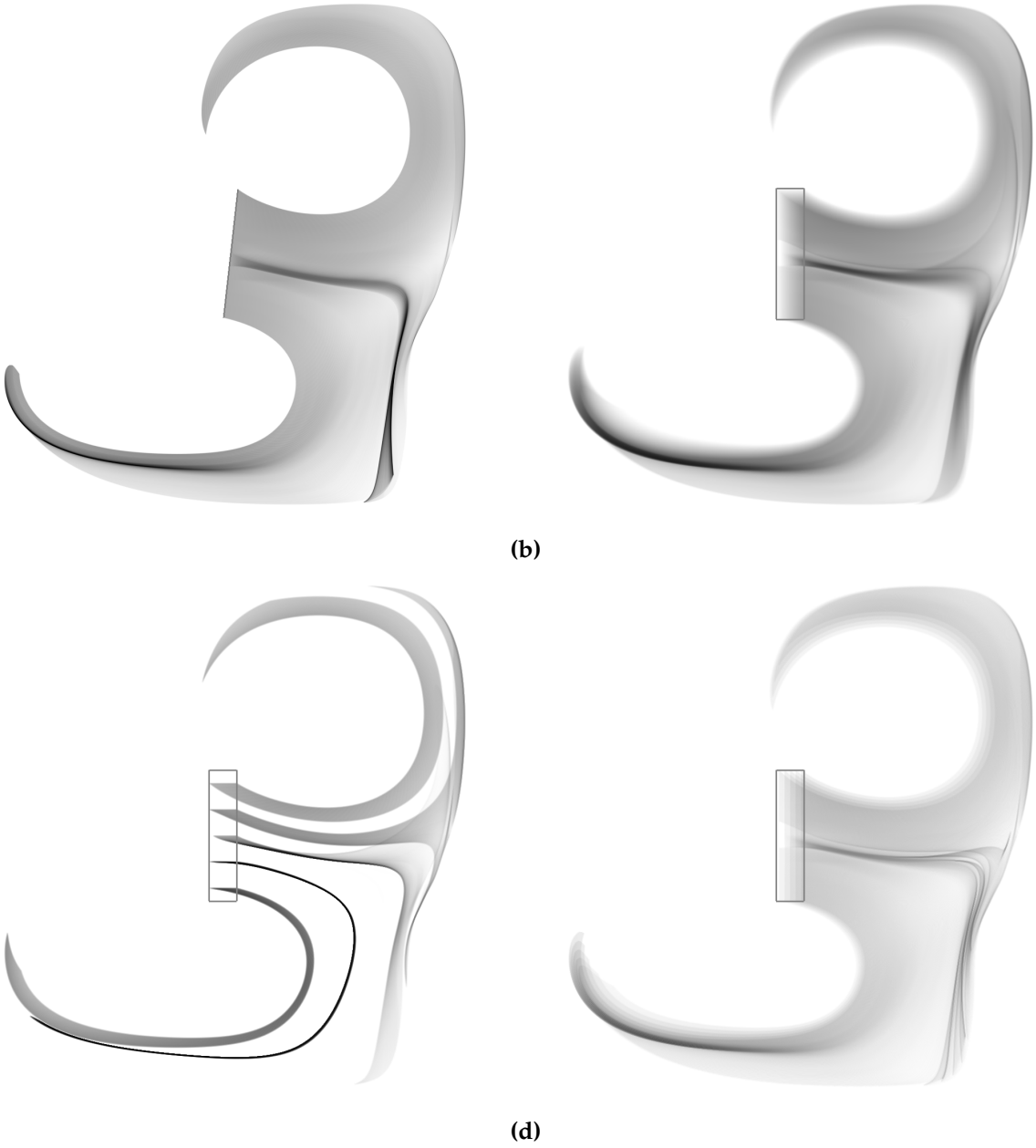
(a) **(b)** **(c)** **(d)**

Abbildung 5.1: Dargestellt wird die Entstehung eines Strömungsbilds zu unterschiedlichen Zeitpunkten. Abbildung (a) und (c) beschreiben die Strömung zu einem früheren Zeitpunkt als (b) und (d). Durch die Darstellung der Partikel in Abbildung (d) kann erkannt werden, dass die Strukturen nicht durch Selbstüberlagerung einer Bahnlinie entstanden sind, sondern dass sich in diesem Bereich mehrere masselose Partikel gesammelt haben.

5.4 Darstellung von Farbverläufen

Für die Analyse der erzeugten Strömungsbilder können die Linien durch Darstellung eines Farbverlaufes die folgenden Zusatzinformationen darstellen.

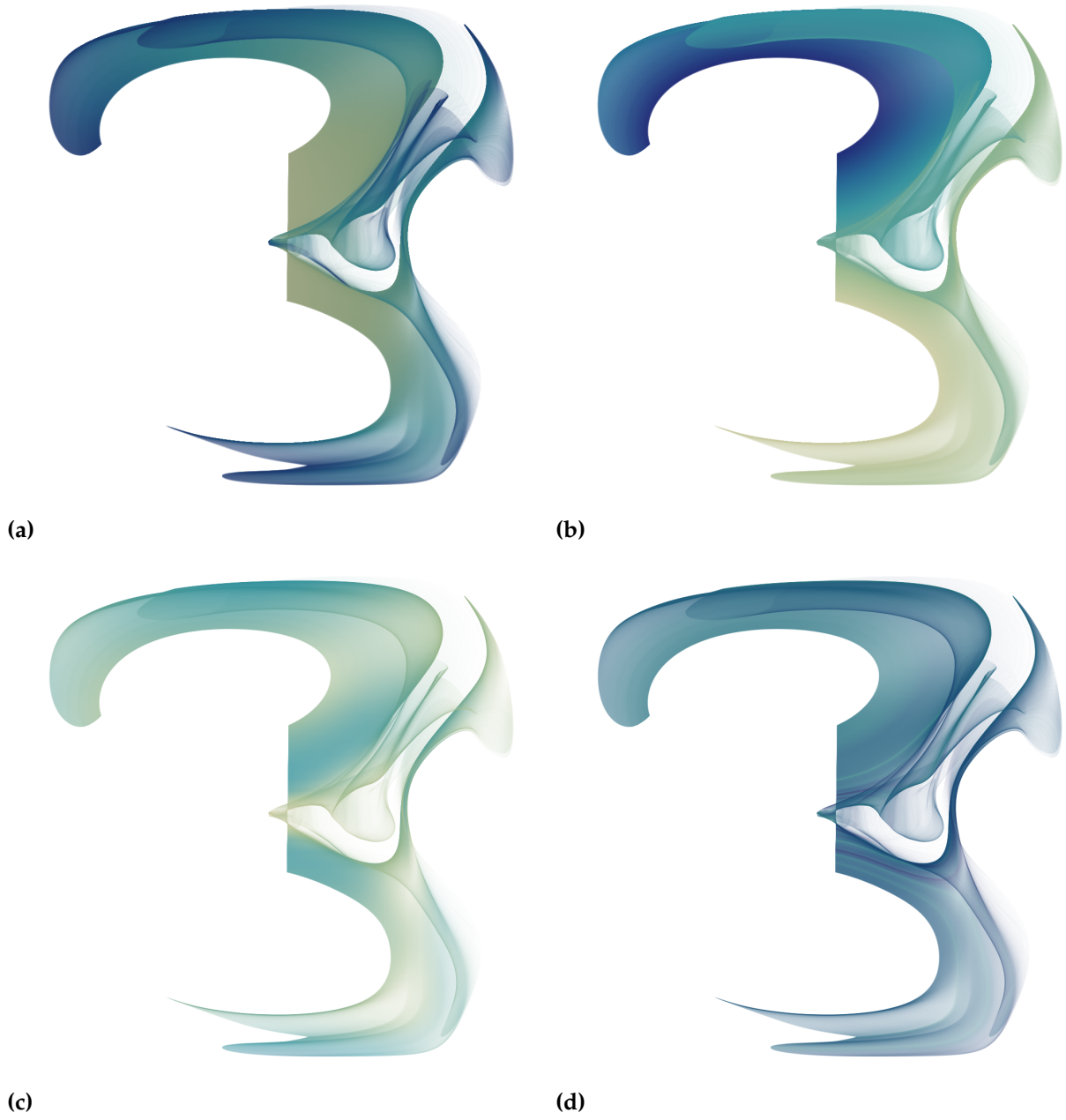
1. Die Integrationszeit in Abbildung 5.3a wird der zeitliche Verlauf der Strömung durch einen linearen Farbverlauf dargestellt.
2. Anhand der Saatposition (siehe Abbildung 5.3b).
3. Anhand der Strömungsgeschwindigkeit (siehe Abbildung 5.3c).
4. Anhand des FTLE-Wertes (siehe Abbildung 5.3d).



(a) **(b)**

(c) **(d)**

Abbildung 5.2: In dieser Abbildung werden verwendete Strategien dargestellt um Saatpunkte zu setzen. In Abbildung (a) werden die Saatpunkte entlang einer Linie gesetzt. In Abbildung (b) gleichmäßig im selektierten Bereich verteilt. In Abbildung (c) wurden 1000*5 und in Abbildung (d) 5*1000 Saatpunkte gesetzt.



(a) **(b)**

(c) **(d)**

Abbildung 5.3: In Abbildung (a) wird der Farbverlauf anhand der Zeit dargestellt. In (b) wird den Linien ein Farbwert durch die Startposition zugewiesen. In (c) wird die Strömungsstärke durch den Farbwert dargestellt und in (d) der FTLE-Wert von der Startposition der Linien.

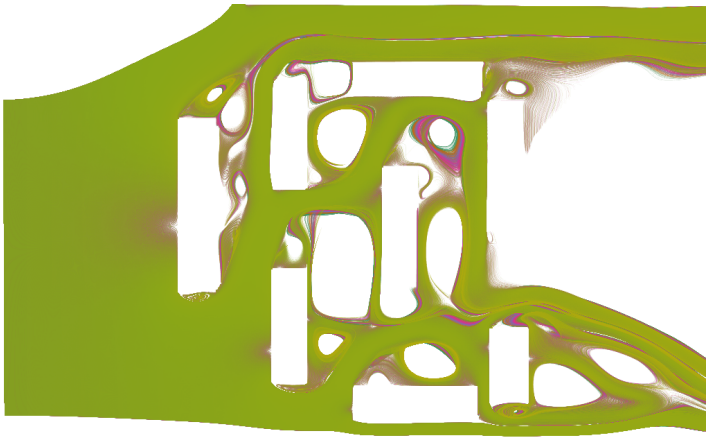


Abbildung 5.4: Kombination von Strömungsbildern zu unterschiedlichen Zeitpunkten.

5.5 Kombination von Resultatbildern

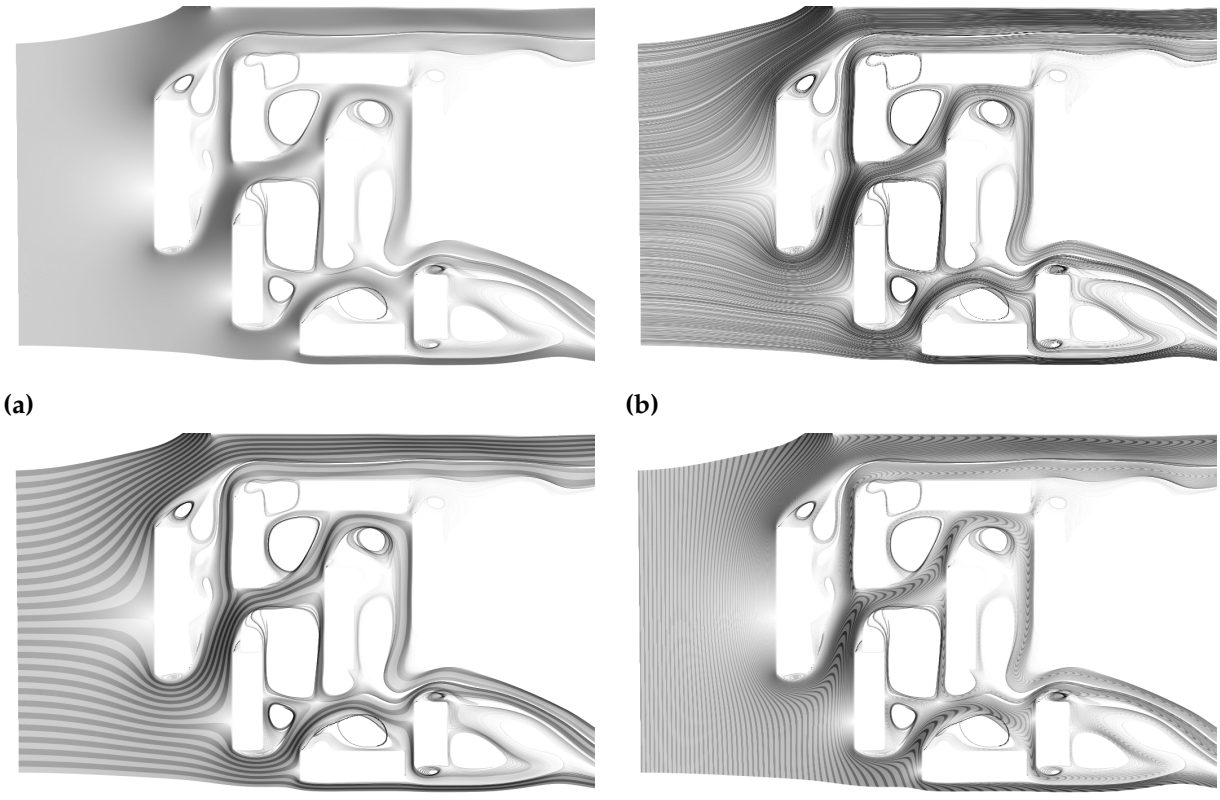
Durch die Kombination von Resultatbildern können zeitabhängig Unterschiede in einer Strömung dargestellt werden. Hierfür werden Strömungsbilder zu unterschiedlichen Startzeitpunkten mit dem identischen Saatbereich erzeugt. Durch das Kombinieren dieser Bilder treten Bereiche der Strömung hervor bei denen es Überlagerungen gibt. Dadurch können Bereiche der Strömung hervorgehoben werden, die über mehrere Zeitbereiche hinweg durchflossen werden.

Durch die Wahl eines geeigneten Farbverlaufes können auch Bereiche sichtbar gemacht werden, die nicht über alle Zeitbereiche hinweg durchflossen werden (siehe Abbildung 5.4). Hierbei wurden die kombinierten Strömungsbilder durch unterschiedliche Farbverläufe eingefärbt. Bereiche, die nicht von anderen Strömungen überlagert werden, werden so durch eine andere Farbe hervorgehoben.

5.6 Verwenden von Strukturen

Um den Strömungsverlauf so genau wie möglich zu beschreiben, könnte man davon ausgehen, dass durch eine größere Menge an verwendeten Bahnlinien der Informationsgehalt steigt oder zumindest konstant bleibt (wenn man zusätzlich die globale Transparenz der Linien anpasst). Jedoch ist dies nicht immer der Fall, da benachbarte Bahnlinien zu einer Fläche verschmelzen können mit der Folge, dass der Verlauf der Strömung nicht mehr erkannt werden kann (siehe Abbildung 5.5a).

Um diesen Effekt zu verhindern, kann den Linien eine Struktur zugewiesen werden, wodurch man die Strömungsrichtung wieder interpretieren kann. Dies wurde in Abbildung 5.5b mit einer zufälligen Einfärbung der Bahnlinien realisiert. In Abbildung 5.5d wurde ein gleichmäßiges Muster verwendet. Bei diesem Muster kann die Frequenz der Linien nicht so



(a) **(b)**

(c) **(d)**

Abbildung 5.5: In den dargestellten Bildern wurden 15000 Saatpunkte in einer Reihe gesetzt. Die Bahnlinien verschmelzen zu einer Fläche in Bild (a). In Bild (b) wurde den Bahnlinien ein zufälliger Transparenzfaktor zwischen [0...1] zugewiesen. In Abbildung (c) wurden gleichmäßige Faktoren verwendet. In (d) wurden bestimmte Zeitbereiche hervorgehoben.

hoch gewählt werden, wie bei dem zufälligen Muster, da es sonst zu Moiré-Mustern kommen würde. Durch das Verwenden einer gleichmäßigen Struktur kann durch die Stauchungen, die an der Struktur erkennbar sind, die Dichte der Linien interpretiert werden (siehe Abbildung 5.5c). In 5.5d wurden die Bahnlinien zu einem festen Zeitintervall strukturiert. Die Strömungsrichtung kann durch die Linienkrümmung erkannt werden. Außerdem kann die Strömungsgeschwindigkeit anhand der Krümmungsintensität abgeschätzt werden.

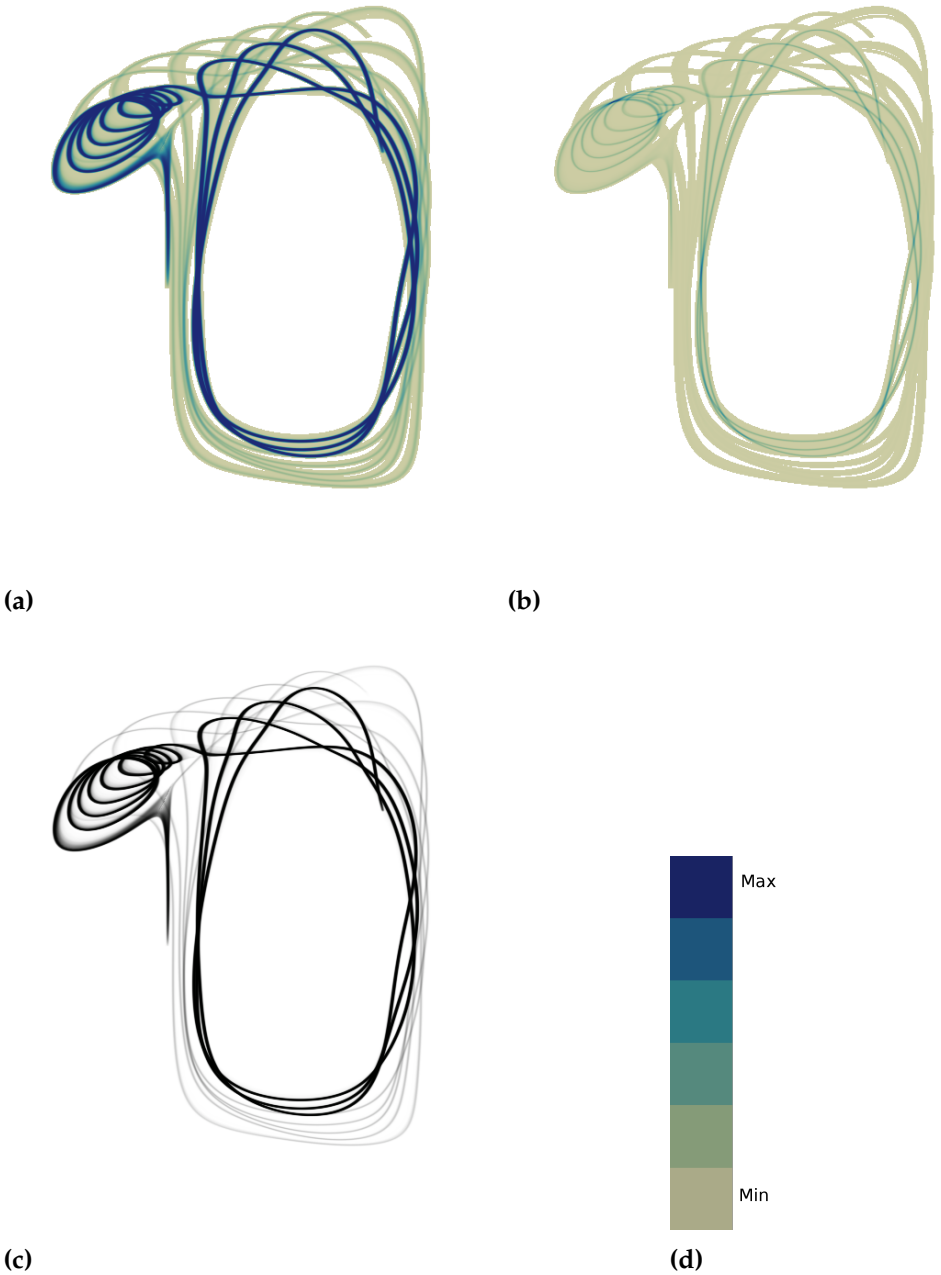
5.7 Bestimmung der Liniendichte

Durch die Darstellung von transparenten Linien kann der Eindruck entstehen, dass anhand der dargestellten Transparenz die Verteilung der Liniendichte genau abgeschätzt werden kann. Jedoch ist das Wachstum des alpha-Werts bei der klassischen Farbmischung nicht linear

und beschränkt. Daher eignet sich die klassische Farbmischung nicht um das tatsächliche Überlagerungsverhältnis darzustellen. Als Alternative kommt eine additive Farbmischung in Frage. Jedoch hat diese Farbmischung den Nachteil, dass kleine Bereiche, an denen sich sehr viele Linien überlagern, das restliche Bild zu transparent erscheinen lassen und somit weniger Information dargestellt werden kann. In Abbildung 5.6a stellt ein Farbverlauf die alpha-Werte eines Strömungsbildes dar, bei dem die klassische Farbmischung verwendet wurde. In Abbildung 5.6b wird der Alpha-Wert durch einen Farbverlauf gezeigt, bei der Additive Farbmischung verwendet wurde und anhand des größten alpha-Wertes im Bild normiert wurde. Abbildung 5.6c zeigt das durch die klassische Farbmischung erzeugte Bild. Vergleicht man Bild 5.6a und 5.6c miteinander sieht man, dass der untere Farbbereich des Farbverlaufes (siehe Abbildung 5.6d) in Abbildung 5.6a nicht mehr sichtbar ist. Dieser Farbbereich ist jedoch in dem Bild, das durch additives Mischen entstanden ist, deutlich höher.

5.8 Anwendung von Blending Techniken

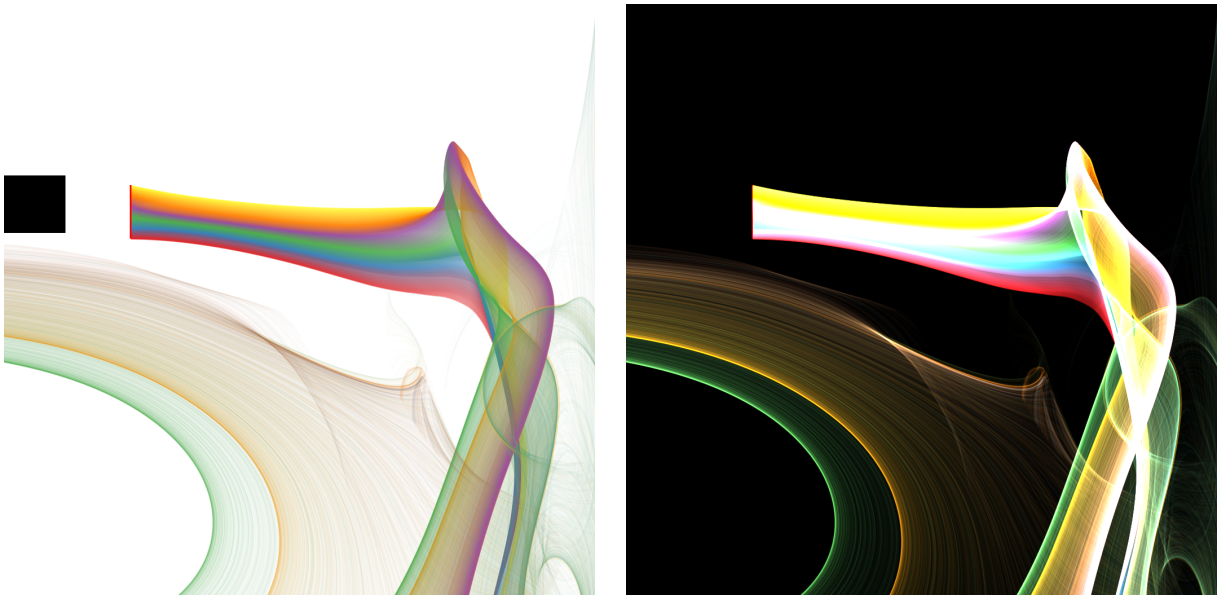
In dieser Arbeit wurden zwei verschiedene Methoden getestet, mit denen die Farbmischung berechnet wurde. Diese Methoden sind zum einen die klassische Farbmischung und zum anderen die additive Farbmischung mit alpha. Diese beiden Methoden werden in Abbildung 5.7 dargestellt. Der Vergleich der beiden Bilder zeigt, dass bei der klassischen Farbmischung die Bereiche, die stärker überlagert werden, weniger transparent sind. Bei der Überlagerung von unterschiedlichen Farben kann erkannt werden, in welcher Reihenfolge die Farben übereinander gezeichnet wurden. Beim additiven Farbmischen wirken die Bereiche heller, in denen sich viele Farben überlagert haben und die Anzahl der sich überlagernden Linien kann genauer interpretiert werden, da die Farbwerte der sich überlagernden Linien gleich gewichtet werden. Jedoch kann nicht erkannt werden, in welcher Reihenfolge die Farben übereinander gezeichnet wurden. Somit eignet sich die klassische Farbmischung besser um den Strömungsverlauf darzustellen, da durch sie die Reihenfolge der Überlagerungen besser dargestellt werden kann.



(a) **(b)**

(c) **(d)**

Abbildung 5.6: In Abbildung (b) wird die Anzahl der Pfadlinien, die durch einen Pixel verlaufen, anhand des Farbverlaufes (d) dargestellt. Im Vergleich wird in Abbildung (a) die Opazität bei normalem alpha-Blending dargestellt. In Abbildung (c) wird das Ergebnisbild durch additives Blending dargestellt. Man kann durch Vergleichen von Bild (c) und (a) erkennen, das der Farbbereich, der keinen Blauanteil enthält, im Bild (a) nicht sichtbar wird. Dieser Farbbereich ist aber überwiegend in Bild (b) vorhanden.



(a) **(b)**

Abbildung 5.7: In den beiden Bildern wird jeweils derselbe Ausschnitt eines Bahnlinienbildes dargestellt. In Abbildung (a) wurde die klassische Farbmischung verwendet und in Abbildung (b) wurde die additive Farbmischung verwendet.

6 Resultate

6.1 Untersuchung der sichtbaren Strukturen

6.1.1 Wahl der Seedingdichte

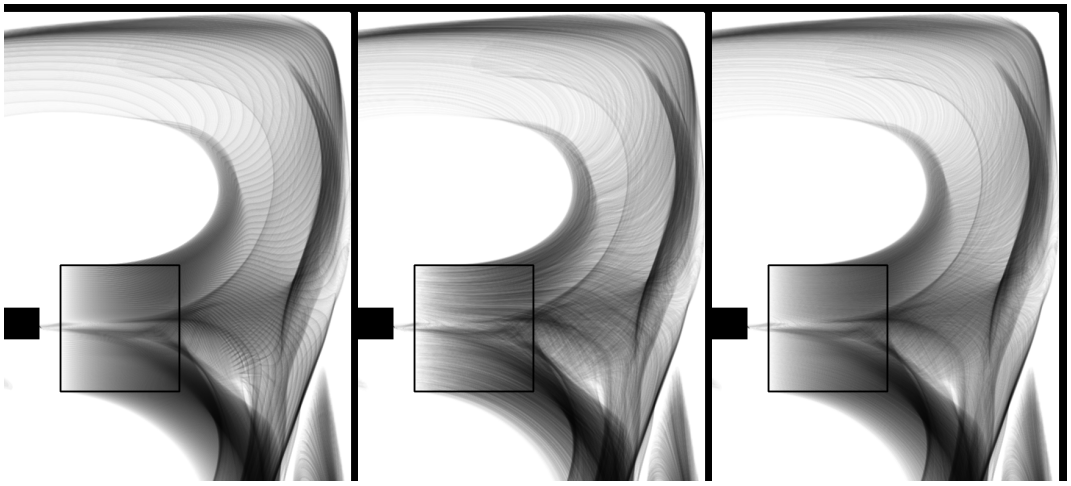
Die in einem Resultatbild sichtbaren Strukturen können Ursachen haben, die nicht direkt etwas mit der Strömungsinformation zu tun haben. Beispielsweise kann sich die Art der Saatpunkt-Verteilung auf die entstehenden Strukturen auswirken. Vergleicht man die Resultate in Abbildung 6.2 miteinander, kann erkannt werden, dass teilweise unterschiedliche Strukturen erkennbar sind, obwohl sowohl derselbe Bereich und Zeitpunkt im Strömungsfeld ausgewählt wurde als auch die gleiche Anzahl an Saatpunkten verwendet wurde.

Da sich die erzeugten Bilder nur in der Art der Saatpunktverteilung unterscheiden, müssen die unterschiedlichen Strukturen auch durch diese verursacht worden sein. In Abbildung 6.1a wurden die Saatpunkte mit gleichmäßigem Abstand zueinander verteilt, wodurch regelmäßige Strukturen in dem Bild erkennbar sind, die in den anderen beiden Bilder nicht auffindbar sind. In Abbildung 6.1b und 6.1c wurden die Saatpunkte zufällig verteilt, es sind in beiden Bildern deutlich mehr Strukturen zu erkennen als in Abbildung 6.1a. Jedoch sind in Abbildung 6.1c weniger Strukturen sichtbar als in 6.1b, da diese mit der "Halton-Folge" erzeugt wurden und dadurch die Saatpunkte gleichmäßiger in Saatbereich verteilt wurden.

Betrachtet man die Strömungsbilder mit dichter Saatpunkt-Verteilung, zeigt sich, dass je dichter die Saatpunkte verteilt sind, umso geringer wird der Unterschied zwischen den unterschiedlichen Saatpunkt Verteilungsstrategien (siehe Abbildung 6.2).

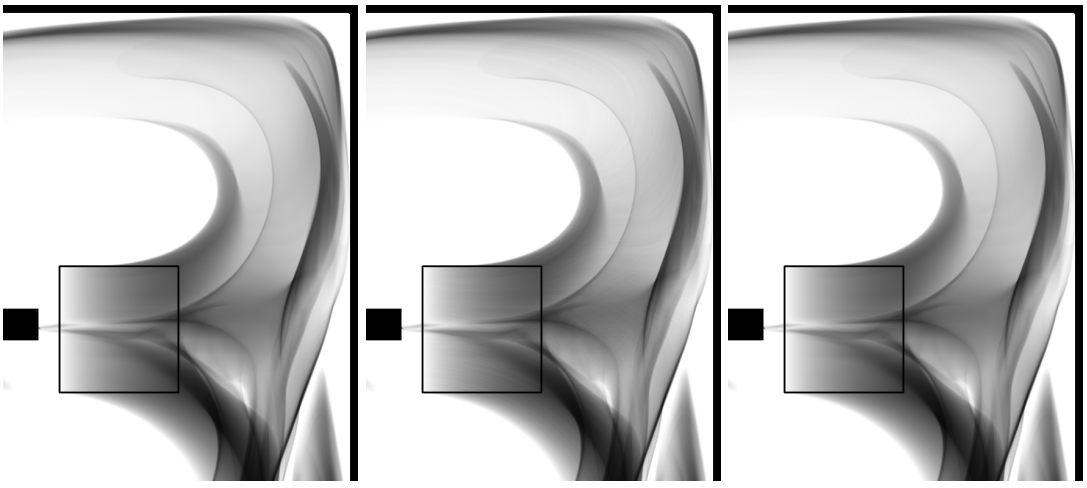
6.1.2 Wahl der Schrittweite

Durch das für die Berechnung der Strömungsbilder verwendete Runge-Kutta-Verfahren kann für den Verlauf der Strömung, bei geeigneter Wahl der Schrittweite, eine sehr hohe Genauigkeit erzielt werden. Andererseits führt die Wahl einer zu großen Schrittweite zu Rechenungenauigkeiten, die deutlich sichtbare Artefakte verursachen. Abbildung 6.3b zeigt stufenartige Strukturen, die entstehen, wenn eine zu große Schrittweite gewählt wurde. In Abbildung 6.3a ist eine geringere Schrittweite gewählt worden, wodurch die Artefakte verschwinden. Betrachtet man die Abbildungen 6.3c und 6.3d, kann erkannt werden, dass die Artefakte durch die große Schrittweite entstanden sind, da sich die Liniensegmente ab einem gewissen Winkel am Anfang zu nahe kommen und am Ende zu weit voneinander entfernt



(a) (b) (c)

Abbildung 6.1: In den selektierten Bereich wurden 150*150 Saatpunkte verteilt. In Abbildung (a) wurden die Saatpunkte mit gleichem Abstand zueinander verteilt. In Abbildung (b) wurden die Saatpunkte zufällig und in Abbildung (c) durch die Haltonsequenz verteilt.



(a) (b) (c)

Abbildung 6.2: In den dargestellten Bildern wurden 1000*1000 Saatpunkte gleichmäßig im selektierten Bereich verteilt. In Abbildung (a) wurden die Saatpunkte mit gleichem Abstand zueinander verteilt. In Abbildung (b) wurden die Punkte zufällig und in (c) durch die Halton-Folge verteilt. Es kann erkannt werden, dass sich die Bilder nicht mehr unterscheiden.

sind. Wird jedoch eine zu geringe Schrittweite gewählt, führt dies zu einem überflüssigen Rechenaufwand.

6.1.3 Analyse über einen Zeitbereich

Zur Analyse verschiedener Bereiche über einen Zeitbereich hinweg wird der "Buoyancy4_2D"-Datensatz verwendet. Dieser enthält die Beschreibung eines Strömungsverlaufs bei geschlossenem Raum, in dem der untere Bereich erwärmt und der obere Bereich abgekühlt wird. Die Auflösung des Datensatzes beträgt $41 * 41$ zu 321 Zeitschritten.

In Abbildung 6.4 werden verschiedene Strukturen, die mit Hilfe des "Buoyancy4_2D"-Datensatz erzeugt wurden, dargestellt. Wie in diesen Abbildungen zu erkennen ist, ist die Strömung sehr unruhig, da schon beim Säen in einem kleinen Bereich komplexe Strukturen entstehen können. In Abbildung 6.4a kann anhand der entstandenen Struktur erkannt werden, dass sich die gesäten masselosen Partikel über einen längeren Zeitraum im unteren Bereich des Strömungsfeldes aufhalten, da sich der untere Bereich erst langsam erwärmt. In Abbildung 6.5b wird im mittleren Bereich des Strömungsfeldes gesät, hier bleiben die masselosen Partikel auch über einen längeren Zeitraum zusammen. Jedoch trennen sich dann die Partikel im oberen Bereich des Strömungsfeldes.

In Abbildung 6.5a wird ein Bereich innerhalb der Strömung untersucht. Hierbei stellen die Bahnlinien mit dem roten Farbverlauf den Weg der ankommenden masselosen Partikel dar und im blauen Verlauf die zukünftige Bahn dieser Partikel.

In Abbildung 6.5b wird eine lange Saatlinie verwendet und es werden rückwärts berechnete Bahnlinien dargestellt. Dadurch zeigt sich, woher die sich momentan auf der Linie befindlichen masselosen Partikel gekommen sind. Zur Sichtbarmachung der Endposition der Bahnlinien wurden die Segmente am Ende der Bahnlinien rot gefärbt und der alpha-Wert auf 1 gesetzt.

Kombination von Zeitschritten

Durch Darstellung eines bestimmten Bereichs innerhalb des Strömungsfeldes zu unterschiedlichen Zeitpunkten kann veranschaulicht werden wie stark sich die Strömung in diesem Bereich verändert. In Abbildung 6.6a, 6.6b und 6.6c wird ein selektierter Bereich zu drei unterschiedlichen Zeitschritten dargestellt. Es ist deutlich zu erkennen, dass sich die Strömung zu den unterschiedlichen Zeitschritten stark verändert. In Abbildung 6.6d wurden diese Informationen in einem Bild kombiniert, dabei wurden die Zeitschritte mit unterschiedlichen Farbverläufen eingefärbt, um sie besser zu unterscheiden.

Betrachtet man Abbildung 6.7c, kann nicht direkt erkannt werden, dass es sich hierbei um eine Überlagerung zeitlich versetzter Bahnlinien handelt. Erst durch die Betrachtung der Zeitsequenzen in Abbildung 6.7b und 6.7c wird dies deutlicher. Eine andere Möglichkeit ist die Verwendung eines Musters, durch das der zeitliche Verlauf dargestellt wird, da somit

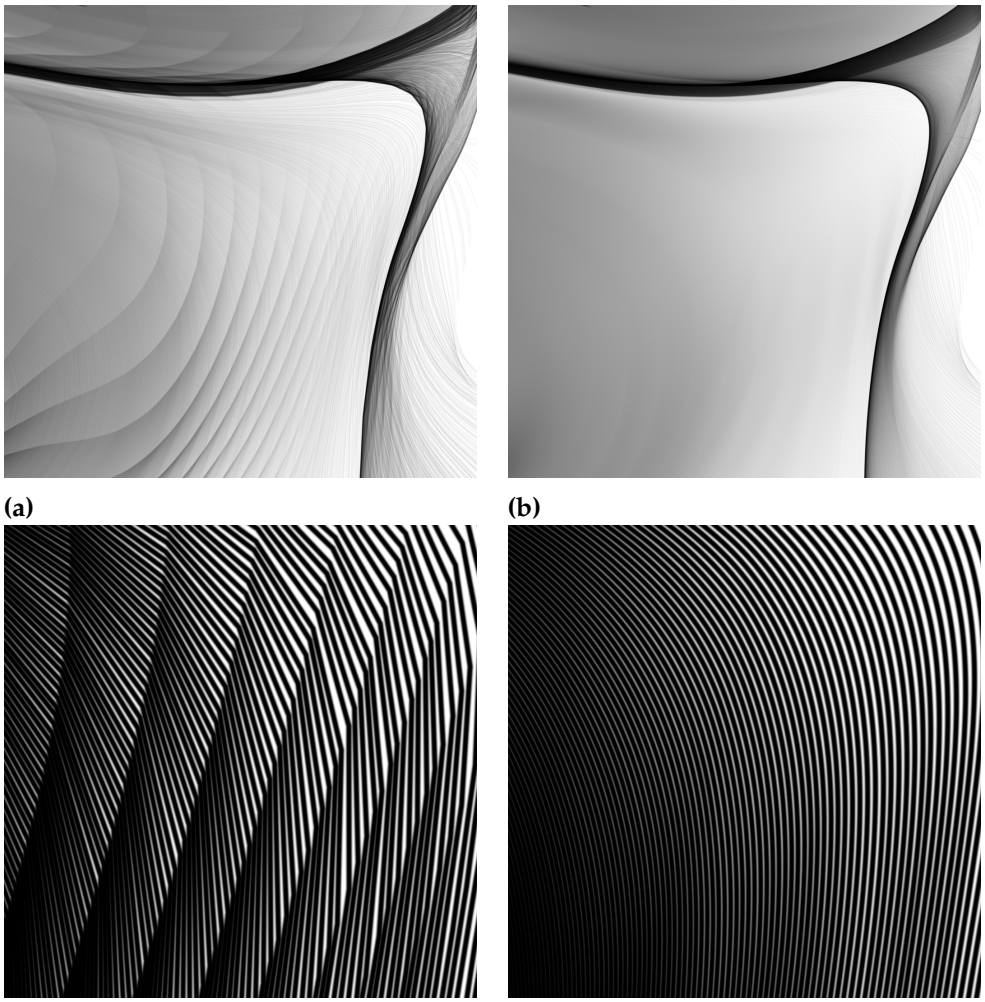
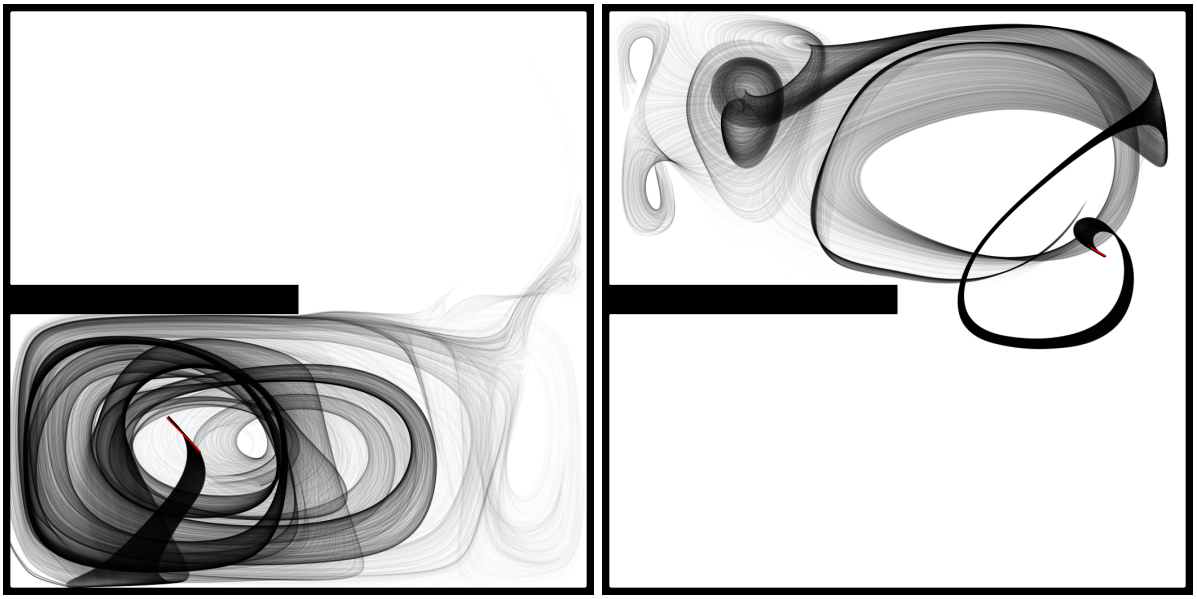


Abbildung 6.3: Abbildung (a) zeigt eine Schrittweite von 0.5 und Abbildung (b) eine Schrittweite von 0.01. Man sieht, dass die dargestellten Strukturen beinahe identisch sind, jedoch ist durch die zu große Schrittweite in Abbildung (a) eine zusätzliche Struktur entstanden. Die Entstehung dieser Strukturen wird in Abbildung (c) und (d) verdeutlicht. In Abbildung (c) wurde eine höhere Schrittweite um ein Liniensegment zu erzeugen gewählt als in Abbildung (d), wodurch Kanten bei der Darstellung der Linien entstehen. Es zeigt sich, dass sich durch die ungenaue Berechnung die Liniensegmente ab einem bestimmten Winkel zunächst zu nahe kommen und sich später weiter voneinander entfernen, verglichen mit den Linien in Abbildung (d), wodurch auch die Artefakte in Abbildung (a) entsandten sein müssen.



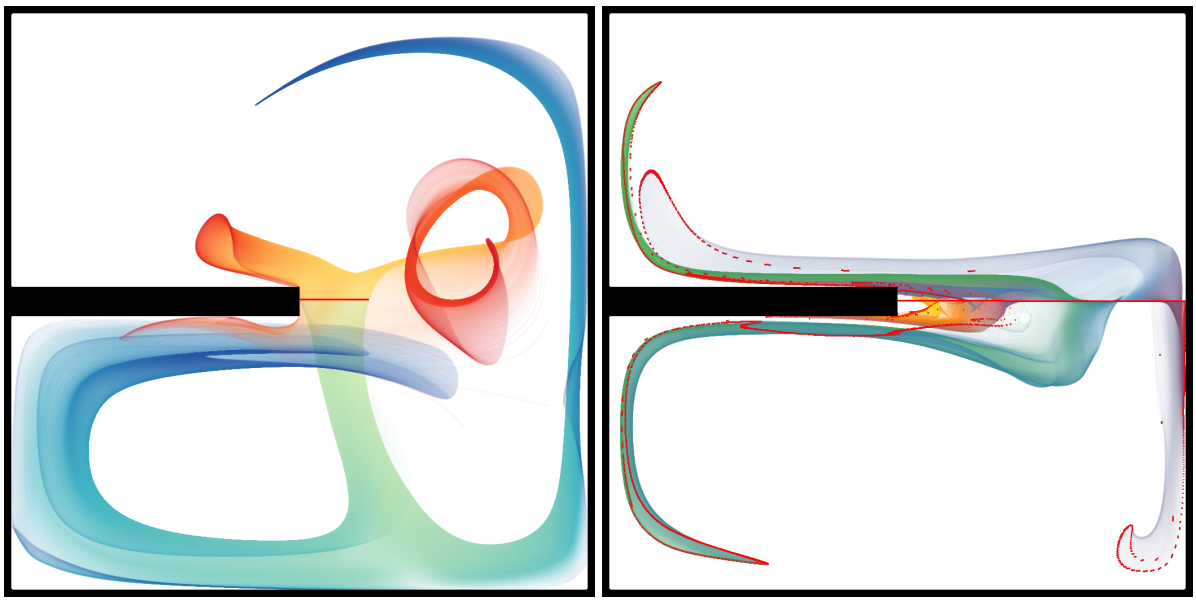
(a) **(b)**

Abbildung 6.4: In Bild (a) und (b) werden Bahnlinien über einen längeren Zeitraum in einer turbulenten Strömung betrachtet. Dabei beginnen die Bahnlinien an der Position 0 und haben eine Integrationsschrittweite von 0.1 und eine Schrittzahl von 700, bei 5000 Bahnlinien, alpha-Wert 0,01.

durch die Transparenz die Zeitlinien beider überlagerter Strukturen sichtbar werden (siehe Abbildung 6.7d).

Untersuchung von FTLE-Werten

Der FTLE-Wert gibt Auskunft über die Separation masseloser Partikel. Somit kann durch einen negativen FTLE-Wert interpretiert werden, dass sich die masselosen Partikel aufeinander zu bewegen. Dies wird in Abbildung 6.8a untersucht. Hierbei wurde in einem Bereich mit negativem FTLE-Wert gesät. Durch die Bahnlinien kann erkannt werden, dass sich die Partikel tatsächlich am Ende der Bahnlinien angenähert haben. In Abbildung 6.8b wurde in einem Bereich gesät, in dem der FTLE-Wert sich dem Wert 0 annähert. Das resultierende Bild zeigt, dass sich die masselosen Partikel in einem Wirbel befinden und sich daher kaum voneinander weg bewegen. In Abbildung 6.8c wurde auf Basis eines relativ hohen FTLE-Werts gesät. Anhand der Bahnlinien zeigt sich, dass der hohe FTLE-Wert durch eine Kollision mit einem Hindernis zustande gekommen ist, da durch die Kollision die masselosen Partikel auseinander gerissen wurden. In Abbildung 6.8d ist ein hoher FTLE-Wert durch den Strömungsverlauf entstanden.



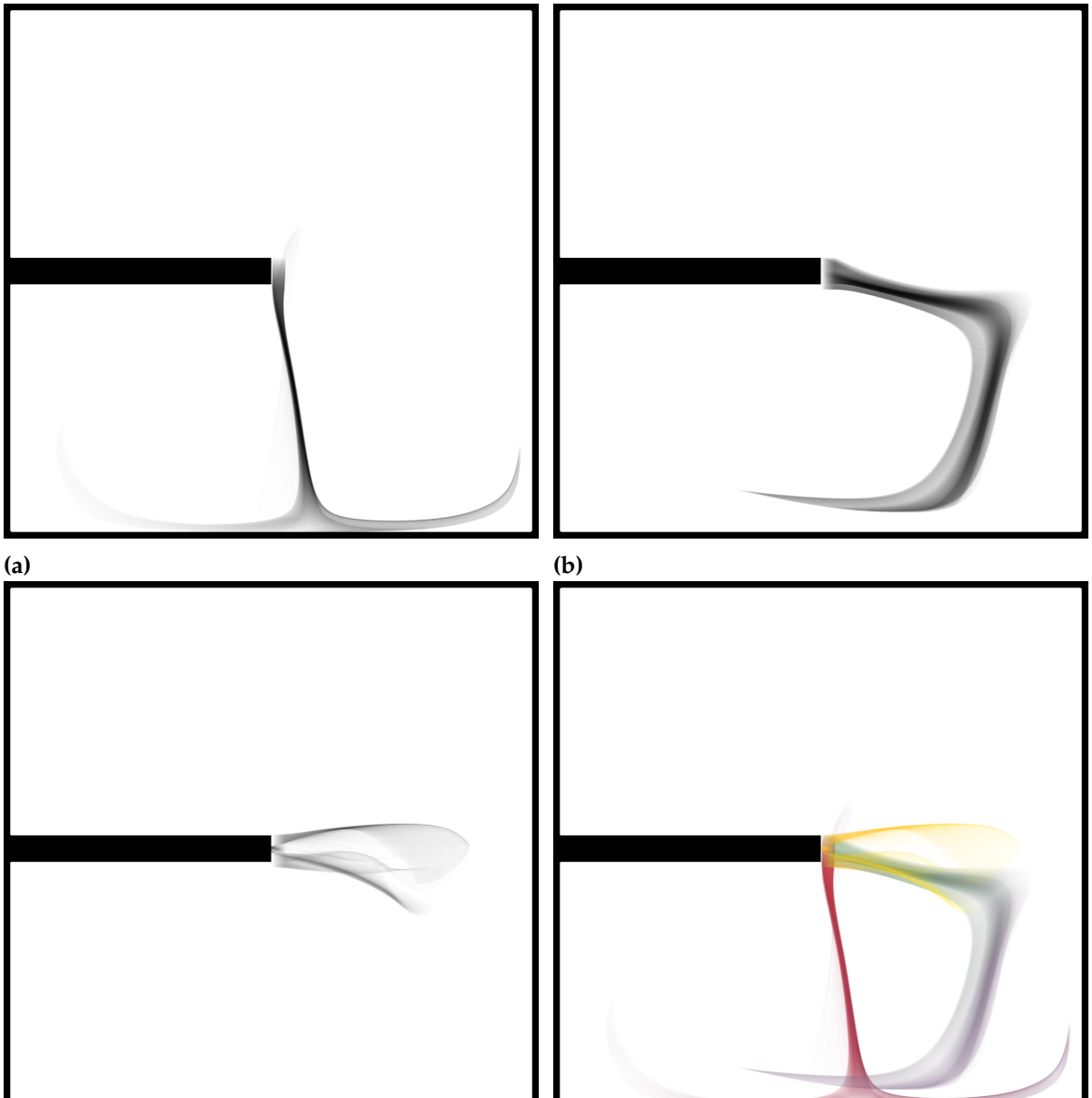
(a) (b)

Abbildung 6.5: In Abbildung (a) wurden die Bahnlinien der ankommenden und weggehenden masselosen Partikel für die rote Linie betrachtet. Startzeitpunkt: 41 von 321, Schrittzahl 150, Bahnlinienzahl 5000, alpha-Wert: 0.01, Schrittweite: für der roten Verlauf -0,1 und für den blauen Verlauf 0.1 In (b) wurden die ankommenden masselosen Partikel für eine selektierte Linie dargestellt. Startzeitpunkt: 164 von 321 Schrittzahl 150 Schrittweite: -0,1, Bahnlinienzahl 5000, alpha-Wert: 0.01

6.1.4 Kármánsche Wirbelstraße

Der "Kármán" Datensatz hat eine Auflösung von 101*301 bei 801 Zeitschritten. Die Strömung verläuft vom unteren zum oberen Bereich des Datensatzes und ist dabei nicht geschlossen. Dabei befindet sich ein Hindernis im mittleren unteren Bereich des Datensatzes.

In Abbildung 6.9 werden Bereiche des Datensatzes untersucht. Dabei werden die Bahnlinien in sieben unterschiedlich farbige Bündel aus Bahnlinien aufgeteilt. Dies hat den Vorteil, dass die Überlagerungen verschiedener Bereiche, die Divergenz der Bahnlinien im Bündel und der ungefähre Linienverlauf leichter zu erkennen sind als bei einer breiten Fläche. In allen Bildern die in Abbildung 6.9 gezeigt werden kann erkannt werden, dass die Bahnlinien aus dem mittleren Bündel das Hindernis auf beiden Seiten umfließen und dann divergieren. Die Abbildungen 6.9a 6.9b und 6.9c beschreiben den Beginn des Datensatzes. Die Bahnlinien aus dem mittleren Bündel umfließen das Hindernis und verhalten sich dann relativ symmetrisch zueinander. Die anderen Bahnlinienbündel werden nur leicht durch das Hindernis abgelenkt, verlaufen dann wieder parallel zueinander. In Abbildung 6.9d verlaufen die beiden Bahnlinienbündel neben dem mittleren Bündel nicht mehr symmetrisch. In Abbildung 6.9e und



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

Abbildung 6.6: Abbildung (a), (b) und (c) stellen denselben Bereich zu unterschiedlichen Zeiten im Strömungsfeld dar. In Abbildung (d) wurden diese Informationen in einem Bild kombiniert. Um dies besser auseinander zu halten, wurde ein Farbverlauf gewählt. Zu den verschiedenen Zeitschritten wurden jeweils 100*100 Bahnlinien erzeugt. Integrationsschrittweite:0,1 Schrittzahl: 100, alpha Wert: 0.005. Die verschiedenen Startzeitpunkte sind((a)23,(b)60),(c)173 von 321 Zeitschritten.

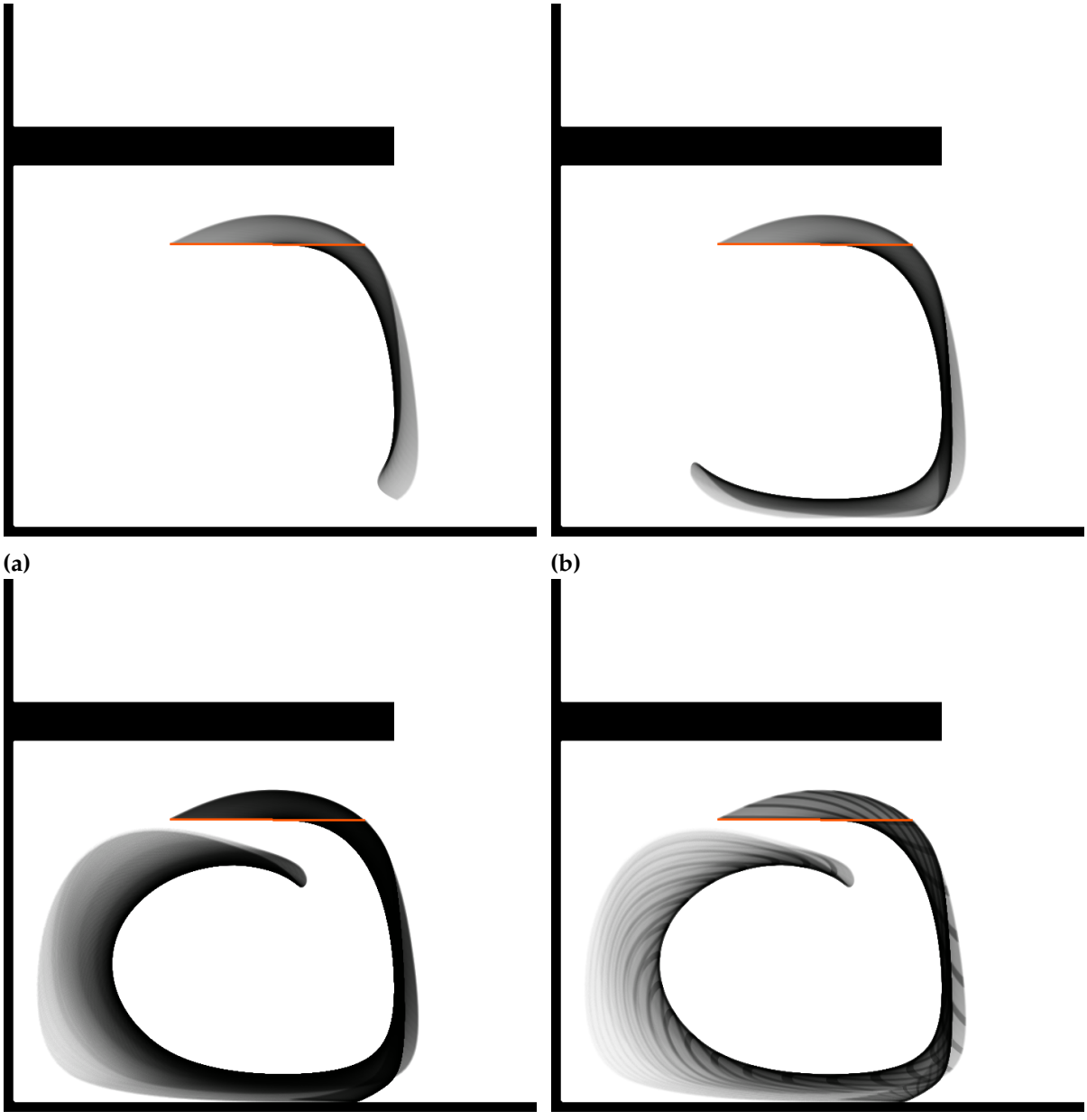
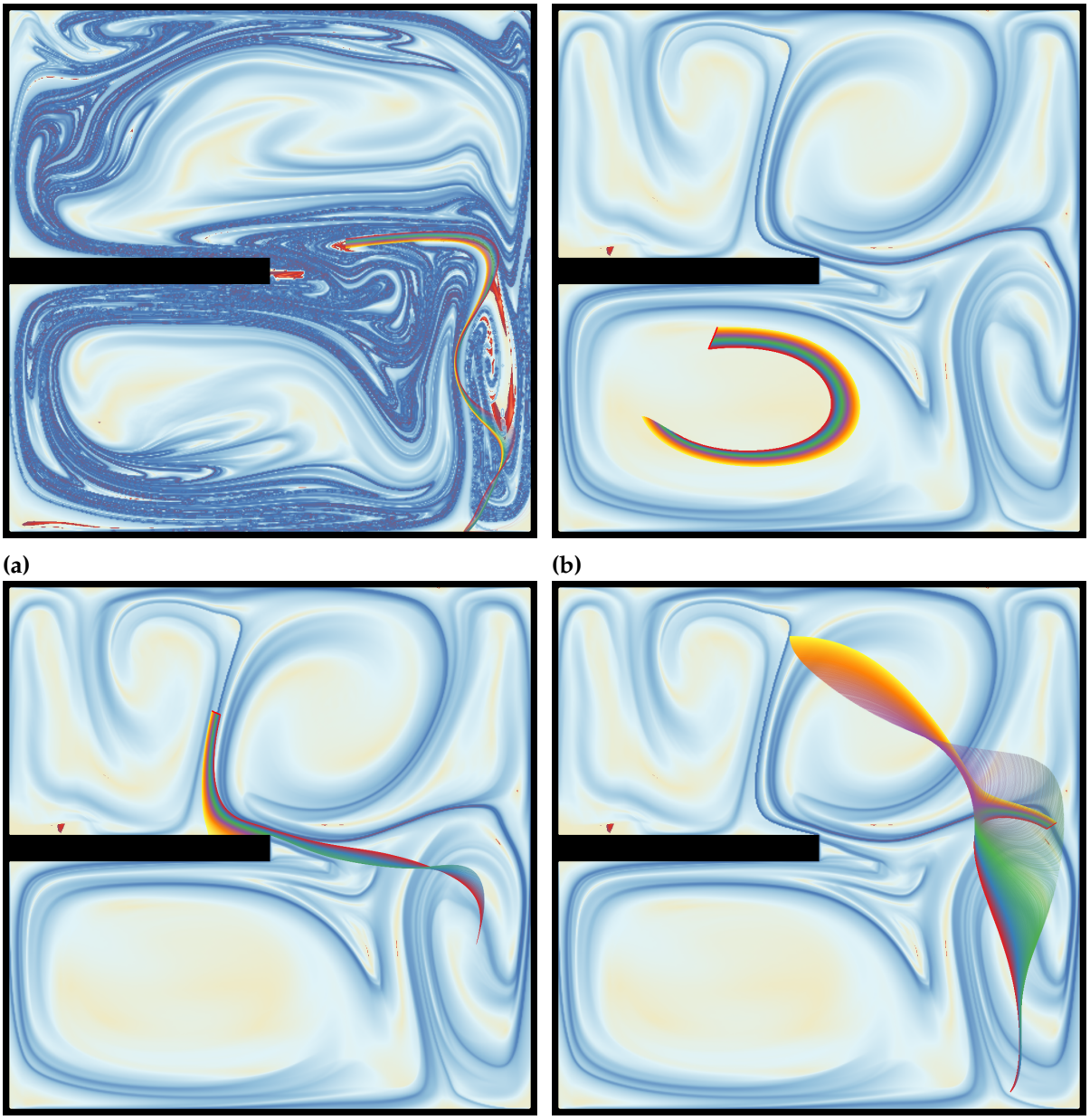


Abbildung 6.7: Abbildung (a),(b) und (c) stellen Sequenzen einer Animation dar. In Abbildung (d) wurde ein Muster verwendet, um den zeitlichen Verlauf darzustellen. Integrationsschrittweite: 0,1, Startzeitpunkt: 26 von 321, Linienanzahl:5500, alpha-Wert: 0,003 Schrittzahl: (a):30,(b):56,(c):84.



(a) (b)

(c) (d)

Abbildung 6.8: In Abbildung (a) wurden Bahnlinien in einem Bereich mit negativem FTLE-Wert gesät. In (b) wird in einem Bereich gesät, dessen FTLE-Wert um 0 herum liegt und in (c) und (d) wird untersucht, wodurch jeweils der relativ hohe FTLE-Wert verursacht wurde. Bei Bild (a) Integrationsschrittweite: 0,1, Startzeitpunkt: 155 von 321, Linienanzahl:500, alpha-Wert: 0,01 Schrittzahl:500. Bei Bild (b),(c)und (d) Integrationsschrittweite: 0,1, Startzeitpunkt: 71 von 321, Linienanzahl:100, alpha-Wert: 0,01 Schrittzahl:500.

6.9f wird gezeigt dass sich danach die Bahnlinienbündel rechts und links vom Hindernis abwechselnd gegenseitig überlagern.

In Abbildung 6.10 wird der Strömungsverlauf anhand von Streichlinien beschrieben. Hierbei kann erkannt werden, dass sich durch das Hindernis im Strömungsfeld zwei gegenläufige Wirbel bilden.

Partikel Endposition

Abbildung 6.11a und 6.11b wurden mit den gleichen Bahnlinien erzeugt. Jedoch wurden in 6.11a im Gegensatz zu 6.11b die Bahnlinien transparent gezeichnet. Durch diesen Unterschied sind Bilder mit unterschiedlichem Informationsgehalt entstanden. In Abbildung 6.11b können durch die fehlende Transparenz Informationen verdeckt worden sein. In Abbildung 6.11a wurde verhindert, dass Informationen durch Verdeckung verloren gehen. Jedoch gehen Informationen, die wenige Bahnlinien betreffen, verloren. In Abbildung 6.11c wurden Informationen aus 6.11b und 6.11a kombiniert. Dadurch zeigt sich, dass Bahnlinien durch die Transparenz unsichtbar gemacht worden sind.

6.1.5 Passieren von Hindernissen

Der "Obstacle" Datensatz hat eine Auflösung von 410*256 bei 64 Zeitschritten. Die Strömung verläuft von links nach rechts und passiert dabei mehrere Hindernisse. Das Strömungsfeld ist von oben und unten begrenzt, hat aber einen kleinen Abfluss an der oberen Begrenzung und ist dabei nicht geschlossen.

Anwendung unterschiedlicher Farbverläufe

In Abbildung 6.13a wurde zu der Strömung der zeitliche Verlauf durch einen Farbverlauf dargestellt. Somit kann abgeschätzt werden, wann welche Strukturen entstanden sind. In Abbildung 6.13b wurde den Bahnlinien anhand ihrer Startposition ein Farbwert zugewiesen. Dies zeigt, von welcher Position aus die Strömung bestimmte Bereiche erreicht. Somit kann erkannt werden, dass die Bahnlinien aus dem grünen und dem blauen Bereich zwischen den Hindernissen hindurchlaufen, während die anders farbigen Bereiche seitlich vorbeilaufen. In Abbildung 6.13c wurden die zusätzlichen Informationen aus Abbildung 6.13b und 6.13a kombiniert. Da zum zeitlichen Verlauf durch Farben, der Strömungsverlauf durch eine zusätzliche zufällige Linienstruktur dargestellt wird. Durch die feinere Struktur kann der Strömungsverlauf genauer dargestellt werden. Sie ist jedoch, verglichen mit Abbildung 6.13c, weniger übersichtlich. In Abbildung 6.13d wird die Strömungsstärke anhand des Farbverlaufes dargestellt.

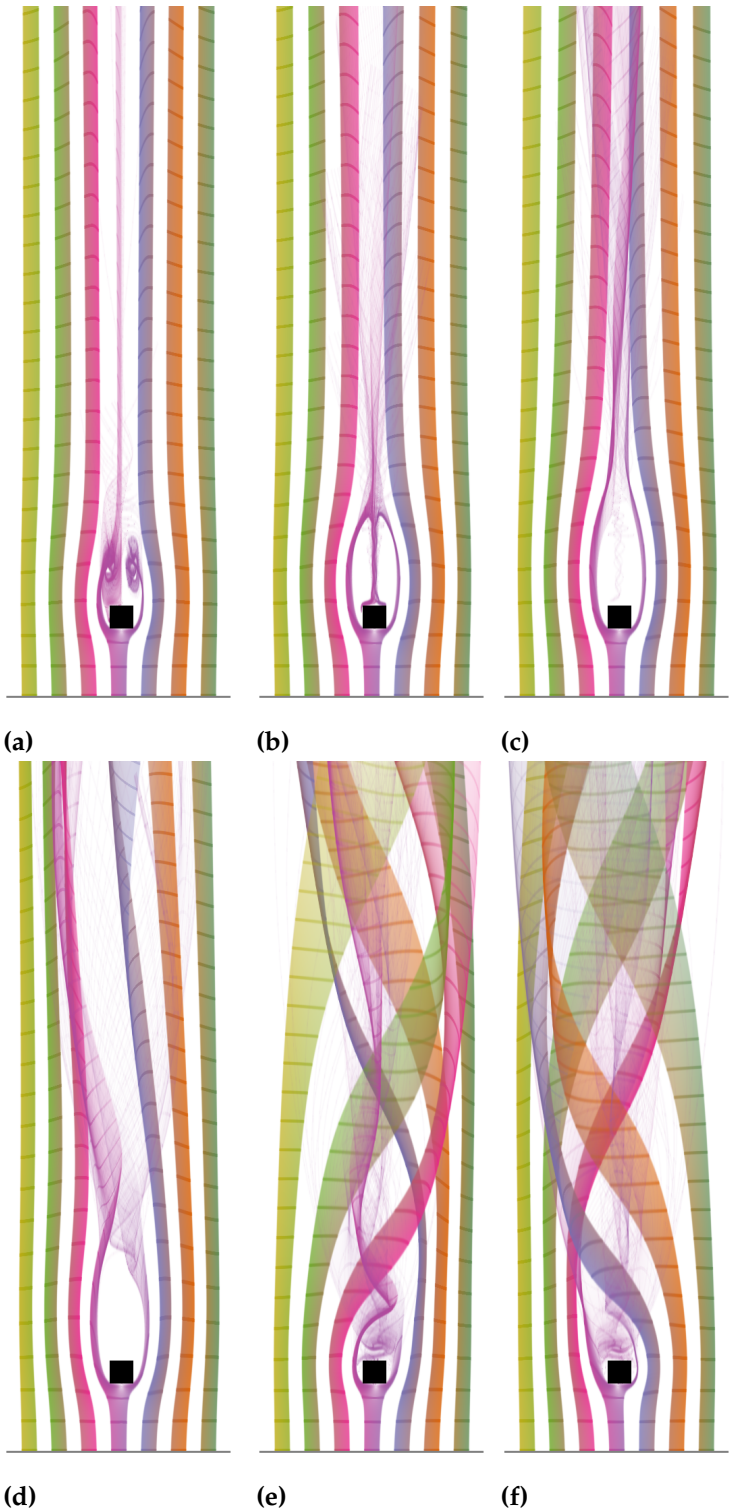


Abbildung 6.9: Resultate zu dem Kármán Datensatz. In den Bildern wird die Strömung durch jeweils sieben Bahnlinienbündel dargestellt. Die verschiedenen Startzeitpunkte ((a): 6,(b): 21,(c): 31,(d): 85,(e):288 und (f): 301 von 801 Zeitschritten). Zusätzlich wurde bei jedem zehnten Liniensegment die Transparenz verdoppelt, um den zeitlichen Verlauf abschätzen zu können. Integrationsschrittweite 0,125 Schrittzahl 350.

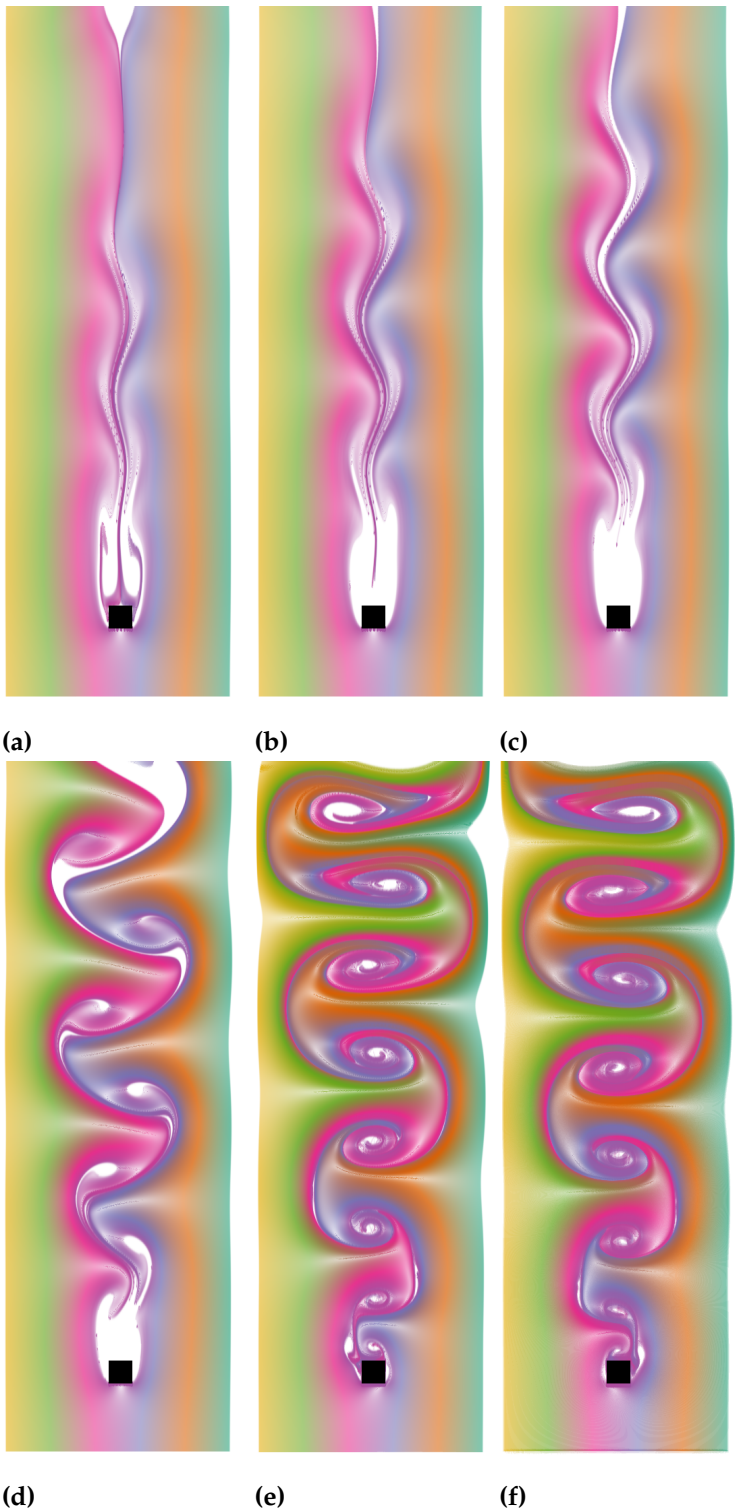
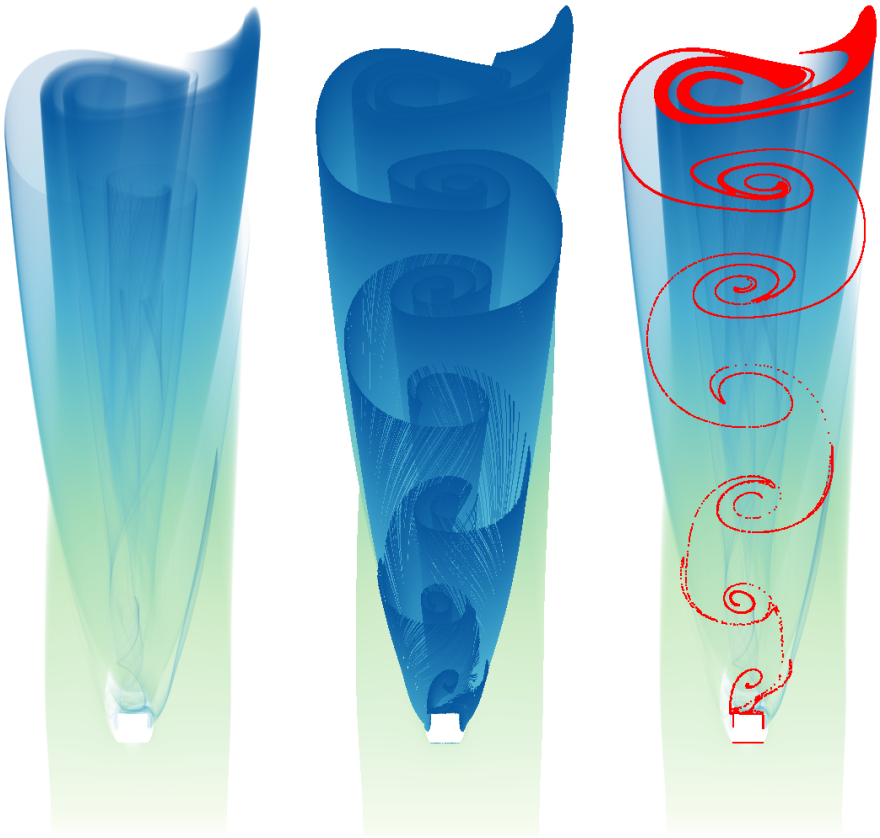


Abbildung 6.10: Resultate zu dem Kármán Datensatz. In den Bildern wird die Strömung durch die Erzeugung von Streichlinien beschrieben. Die verschiedenen Startzeitpunkte (a): 6,(b): 21,(c): 31,(d): 85,(e):288 und (f): 301 von 801 zeitschritten, Partikelerzeugungsrate: 1200 mal jeden 0,1 ten Zeitschritt

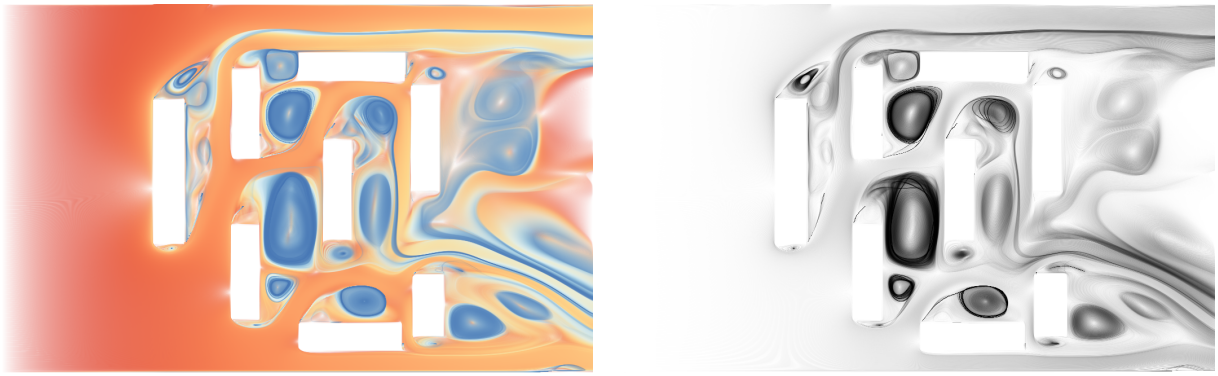


(a) (b) (c)

Abbildung 6.11: Abbildung (a) stellt eine Strömung dar, die durch die Überlagerung von transparenten Bahnlinien sichtbar gemacht wurde. In Abbildung (b) wurden undurchsichtige Bahnlinien verwendet. Die beiden Bilder enthalten unterschiedliche Informationen. In Abbildung (c) wurden diese Informationen zusammengeführt. Integrationsschrittweite: 0,5, Startzeitpunkt: 299 von 801, Linienanzahl:1000*1000, Schrittzahl:200 alpha-Wert: bei (b): 1 und bei (a) 0,0005.

Untersuchung von Details

Durch die Erzeugung sehr langer Bahnlinien können Strukturen wie Wirbel bei einer nicht geschlossenen Strömung sichtbar gemacht werden. Dabei bewegen sich die masselosen Partikel, durch deren Verlauf die Linien gezeichnet werden , länger in der Strömung. In Abbildung (b) wurde ein zeitlich ansteigender Alpha Wert verwendet um die älteren Partikelverlaufe zu identifizieren. In Abbildung (a) wurde ein Farbverlauf gewählt, der einen starken Kontrast zwischen der Start- und der Ende-Farbe aufweist.In Abbildung (c) wird dargestellt, aus welchem Bereich die imaginären Partikel kommen die zu einem festgelegten Zeitraum die rote Linie berühren

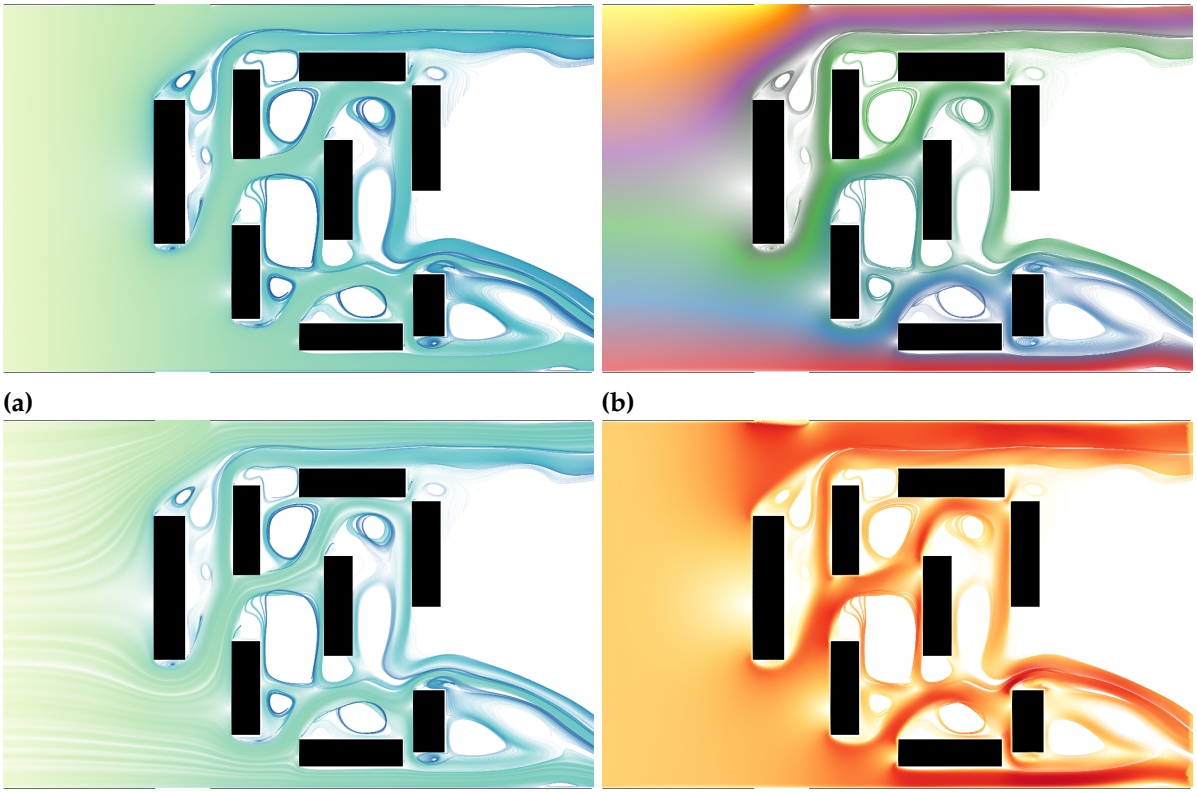


(a) (b)

(c)

Abbildung 6.12: Integrationsschrittweite: bei (a),(b) 0,3,(c) -0,1 Startzeitpunkt: bei (a),(b) 0,(c) 60 von 64, Linienanzahl: bei (a), (b) 1000*1000, (c) 1500, Schrittzahl: bei (a) 121,(b) 201, (c) 400, alpha-Wert: bei (a)0,0005 bei (b) min-alpha:0 max-alpha:1, (a)0.03

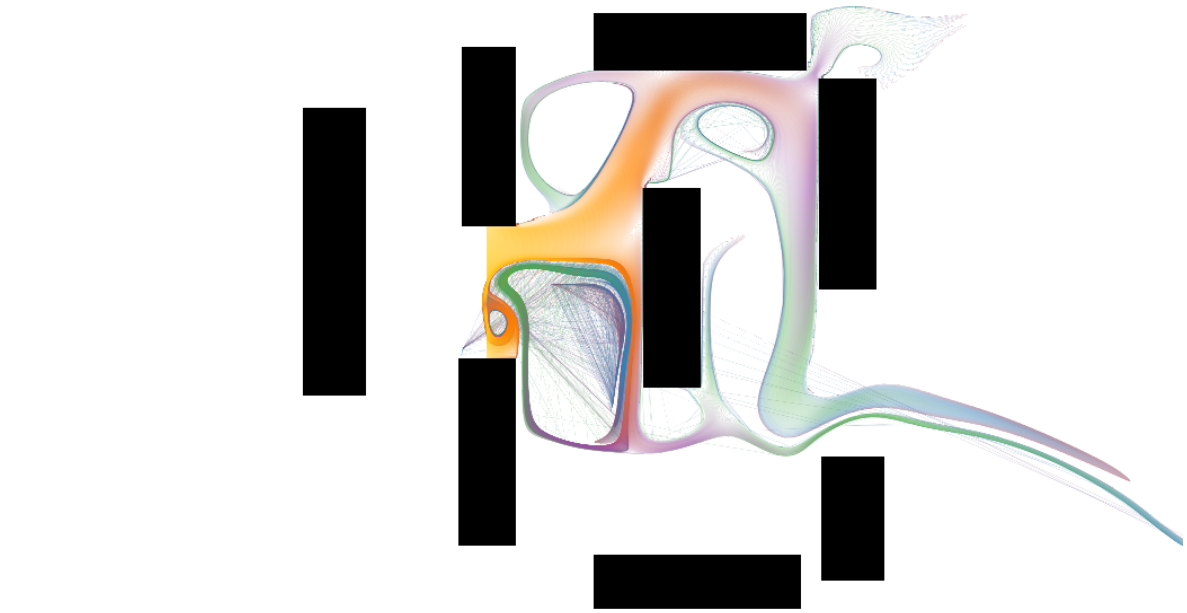
Bei der Darstellung von Streichlinien gibt es das Problem, dass durch die Wahl eines zu großen Zeitintervalls beim Setzten der masselosen Partikels an die Saatposition, zu große Abstände zwischen den Partikeln entstehen, und somit der Störungsverlauf nicht mehr genau dargestellt werden kann. In Abbildung 6.14 wird dies dargestellt.



(a) **(b)**

(c) **(d)**

Abbildung 6.13: In Abbildung (a) wurde der zeitliche Verlauf durch einen Farbverlauf dargestellt, die Strömung verläuft von weiß nach dunkelblau. In Abbildung (b) wurde den Bahnlinien eine Farbe anhand ihrer Startposition zugewiesen. In Abbildung (c) wurde der zeitliche Verlauf durch einen Farbverlauf und der Strömungsverlauf durch eine zufällige Linienstruktur dargestellt. In (d) wurde die Strömungsgeschwindigkeit durch Farben von weiß nach rot kodiert. Bei den Bildern wurden jeweils 5000 Linien am linken Rand zum Zeitpunkt 0 von 64 für 500 Integrationsschritte gesät. Die Schrittweite betrug 0,1 und der alpha-Wert 0.2



(a)

Abbildung 6.14: Darstellung von Streichlinien mir einem zu großen zeitlichen Abstand beim setzten der masselosen Partikel an die Saatpositionen.

7 Zusammenfassung und Ausblick

Diese Arbeit hat die beiden folgenden Ziele: Es soll herausgefunden werden, ob die Darstellung sehr vieler charakteristischer Linien durch geeignete Überlagerungstechniken verbessert werden kann und ob die daraus resultierenden Bilder neuen Einblick in die Strömungsdaten erlauben.

Während dieser Arbeit wurde ein Programm entwickelt, das die Möglichkeit bietet, eine hohe Anzahl von charakteristischen Linien auf der GPU zu berechnen. Dabei wurde eine Technik angewendet, die es ermöglicht, die Linien zeitlich korrekt zu überlagern, auch wenn die Linien gleichzeitig in einer instationären Strömung entstanden sind. Dabei müssen die Linien jedoch nicht am Stück auf der GPU berechnet werden, was einen enormen Speicheraufwand zur Folge hätte, sondern werden schrittweise erzeugt. Somit ist der benötigte Speicherplatz unabhängig von der Länge der charakteristischen Linien.

Das so erzeugte Linienbild wurde untersucht und Fehler, die durch Rasterisierungsprobleme entstanden sind, wurden identifiziert und durch Anwendung geeigneter Techniken behoben.

Dabei wurden zu dem angewendeten additiven- und alpha- Blending auch zeitlich ansteigende und abfallende alpha-Werte verwendet.

Danach wurden durch Anwendung verschiedener Saattechniken Rückschlüsse auf die Entstehung von Strukturen im Strömungsbild gezogen und durch das Einbeziehen von Mustern wurde untersucht, ob der Informationsgehalt gesteigert werden kann. Zum Schluss wurde durch Berechnen des Separationswertes von masselosen Partikeln eine gewichtete Saattechnik entwickelt, um die Bahnlinien bei divergenten Strömungen gleichmäßig im divergierenden Bereich zu verteilen.

Zusätzlich zu den Saattechniken wurden verschiedene Farbkodierungen getestet um bessere Rückschlüsse auf die entstandenen Strukturen ziehen zu können. Zudem wurde die Möglichkeit geboten, automatisch Serienbilder zu generieren um Animationen zu erzeugen und um die Suche nach interessanten Bereichen innerhalb der Strömung zu erleichtern.

Danach wurden die erzeugten Bahnlinien mit FTLE-Werten im Saatbereich verglichen, um die Entstehung der FTLE-Werte zu visualisieren.

Durch die in dieser Arbeit entwickelte Technik konnte gezeigt werden, dass es möglich ist mit Hilfe von Transparenz eine sehr hohe Anzahl an Bahn- und Pfadlinien zu überlagern, um somit ein aussagekräftiges Strömungsbild zu erzeugen. Dabei verschmelzen die Linien ab einer bestimmten Dichte optisch zu einer glatten transparenten Fläche. Durch die unterschiedlichen transparenten Bereiche innerhalb der Fläche können Regionen erkannt werden,

in denen sich die Linien zu unterschiedlichen Zeiten überlagern oder dichter zusammenlaufen. Zusätzlich konnten weitere Informationen durch Strukturierung und Einfärbung der einzelnen Linien dargestellt werden, um die Entstehung der dargestellten Strukturen zu visualisieren.

Problematisch bei dieser Technik ist jedoch, dass Strukturen zu transparent oder zu stark überlagert sein können und dadurch nicht sichtbar werden.

In dieser Arbeit wurden Überlagerungen von Bahn- und Streichlinien untersucht. Bei Streichlinien ist der Rechenaufwand jedoch deutlich höher, da sie eine Verbindungslinie zwischen masselosen Partikeln darstellen, die hintereinander an der selben Saatposition zu unterschiedlichen Zeitpunkten erzeugt worden sind. Bei Bahnlinien hingegen muss für jeden Saatpunkt nur eine Partikelbahn berechnet werden. Das Erzeugen neuer Partikel für die Streichlinie muss in einem sehr geringen zeitlichen Abstand erfolgen, da sonst zu lange Liniensegmente entstehen und somit der Strömungsverlauf zu ungenau beschrieben wird.

Eine Möglichkeit könnte sein, bei zu langen Liniensegmenten neue masselose Partikel zu erzeugen, um mit ihnen den Strömungsverlauf genauer zu beschreiben. Dabei müssten die Zeitpunkte an denen die Partikel, durch die das zu lange Liniensegment beschrieben wird, am Saatpunkt gestartet sind ermittelt werden und die neuen Partikel zwischen diesen Zeitpunkten am Saatpunkt ausgesetzt werden.

Literaturverzeichnis

[BM08] I. Bronštejn, G. Musiol. *Taschenbuch der Mathematik*. Deutsch, 2008. URL http://books.google.de/books?id=VnsL9p8hXfQC. (Zitiert auf Seite 16)

[BPR01] D. Bürkle, T. Preußer, M. Rumpf. Transport and Anisotropic Diffusion in Time-dependent Flow Visualization. In *Proceedings of the Conference on Visualization '01*, VIS '01, S. 61–68. IEEE Computer Society, Washington, DC, USA, 2001. URL http://dl.acm.org/citation.cfm?id=601671.601679. (Zitiert auf Seite 12)

[BR98] J. Becker, M. Rumpf. *Visualization of Time-dependent Velocity Fields by Texture Transport*. Preprint. Albert-Ludwigs-Univ., Math. Fak., 1998. URL http://books.google.de/books?id=ghiRtwAACAAJ. (Zitiert auf Seite 12)

[Cor13] N. Corporation. *NVIDIA CUDA C Programming Guide*, 2013. (Zitiert auf den Seiten 20 und 21)

[Cyn14] COLORBREWER 2.0, 2014. URL www.colorbrewer2.org/. (Zitiert auf Seite 24)

[DB07] R. M. T. I. David Borland. Rainbow Color Map (Still) Considered Harmful. *IEEE Comput. Graph. Appl.*, 27(2):14–17, 2007. doi:10.1109/MCG.2007.323435. URL http://dx.doi.org/10.1109/MCG.2007.323435. (Zitiert auf Seite 23)

[HA09] P. M. Hansjörg Albrecht, Andreas Binder. *Einführung in die Finanzmathematik*. Martin Brokate, Heinz W. Engl, Karl-Heinz Hoffmann, Götz Kersting, Gernot Stroth, Emo Welzl, 2009. (Zitiert auf Seite 23)

[HH89] J. L. Helman, L. Hesselink. Representation and Display of Vector Field Topology in Fluid Flow Data Sets. *Computer*, 22(8):27–36, 1989. doi:10.1109/2.35197. URL http://dx.doi.org/10.1109/2.35197. (Zitiert auf Seite 13)

[HTS01] H. Hagen, X. Trichoche, G. Scheuermann. Topology-Based Visualization of Time-Dependent 2D Vector Fields. In *Data Visualization 2001 (Proceedings of the Joint Eurographics - IEEE TCVG Symposium on Visualization in Ascona, Switzerland), Ebert, Favre, Peikert (Hrsg.), Springer, Wien,2001,*, S. 117–126. 2001. (Zitiert auf Seite 13)

[JEH01] B. Jobard, G. Erlebacher, M. Y. Hussaini. Lagrangian-Eulerian Advection for Unsteady Flow Visualization. In *Proceedings of the Conference on Visualization '01*, VIS '01, S. 53–60. IEEE Computer Society, Washington, DC, USA, 2001. URL http://dl.acm.org/citation.cfm?id=601671.601678. (Zitiert auf Seite 12)

[JL00] B. Jobard, W. Lefer. Unsteady Flow Visualization by Animating Evenly-Spaced Streamlines. *Comput. Graph. Forum*, 19(3):31–39, 2000. URL http://dblp.uni-trier.de/db/journals/cgf/cgf19.html#JobardL00. (Zitiert auf Seite 12)

[KLG+13] A. Kuhn, N. Lindow, T. Günther, A. Wiebel, H. Theisel, H.-C. Hege. Trajectory Density Projection for Vector Field Visualization. *EuroVis - Short Papers 2013*, S. 31–35, 2013. (Zitiert auf Seite 13)

[LH04] R. Laramee, H. Hauser. The State of the Art in Flow Visualization: Dense and Texture-Based Techniques. *Published in the Journal CGF -Computer Graphics Forum*, 23(2):pp. 203–233, 2004. URL http://www.vrvis.at/publications/PB-VRVis-2004-008. (Zitiert auf Seite 11)

[LHD+04] R. S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F. H. Post, D. Weiskopf. The state of the art in flow visualization: Dense and texture-based techniques. *Computer Graphics Forum*, 23:2004, 2004. (Zitiert auf den Seiten 11 und 15)

[LHZP07] R. S. Laramee, H. Hauser, L. Zhao, F. H. Post. Topology-based flow visualization, the state of the art. In H. Hauser, H. Hagen, H. Theisel, Herausgeber, *Topology-Based Methods in Visualization*. Springer Verlag, Mathematics and Visualization Series, 2007. URL http://graphics.tudelft.nl/Publications-new/2007/LHZP07a. (note 978-3-540-70822-3). (Zitiert auf Seite 13)

[LJH03] R. S. Laramee, B. Jobard, H. Hauser. Image Space Based Visualization of Unsteady Flow on Surfaces. In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, VIS '03, S. 18–. IEEE Computer Society, Washington, DC, USA, 2003. doi:10.1109/VISUAL.2003.1250364. URL http://dx.doi.org/10.1109/VISUAL.2003.1250364. (Zitiert auf Seite 12)

[LL99] W. d. Leeuw, R. v. Liere. Spotting Structure in Complex Time Dependent Flow. In *Dagstuhl '97, Scientific Visualization*, S. 47–53. IEEE Computer Society, Washington, DC, USA, 1999. URL http://dl.acm.org/citation.cfm?id=647367.723597. (Zitiert auf Seite 11)

[LMJI02] Z. Liu, R. J. Moorhead, R. James, M. Ii. AUFLIC: An Accelerated Algorithm For Unsteady Flow Line Integral Convolution, 2002. (Zitiert auf Seite 11)

[MB96] N. Max, B. Becker. Flow Visualization Using Moving Textures. In D. C. Banks, T. W. Crockett, K. Stacy, Herausgeber, *Proceedings of the ICAS/LaRC Symposium on Visualizing Time-Varying Data*, NASA Conference Publication 3321, S. 77–87. 1996. (Zitiert auf Seite 12)

[MLP+09] T. McLoughlin, R. S. Laramee, R. Peikert, F. H. Post, M. Chen. Over Two Decades of Integration-Based, Geometric Flow, 2009. (Zitiert auf Seite 12)

[NFHS11] A. Nischwitz, M. Fischer, P. Haberäcker, G. Socher. *Computergrafik und Bildverarbeitung*. Vieweg Verlag, Friedr, & Sohn Verlagsgesellschaft mbH, 2011. URL http://books.google.de/books?id=LNCjfPfx7QkC. (Zitiert auf den Seiten 24, 25, 26, 27 und 28)

[PD84] T. Porter, T. Duff. Compositing Digital Images. *SIGGRAPH Comput. Graph.*, 18(3):253–259, 1984. doi:10.1145/964965.808606. URL http://doi.acm.org/10.1145/964965.808606. (Zitiert auf Seite 25)

[SA07] J. H. Spurk, N. Aksel. Strömungslehre Einführung in die Theorie der Strömungen, 2007. URL http://www.worldcat.org/search?qt=worldcat_org_all&q=3540384391. (Zitiert auf den Seiten 15 und 16)

[SHH+07] M. Schlemmer, I. Hotz, B. Hamann, F. Morr, H. Hagen. Priority Streamlines: A Context-based Visualization of Flow Fields. In *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization*, EUROVIS'07, S. 227–234. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007. doi:10.2312/VisSym/EuroVis07/227-234. URL http://dx.doi.org/10.2312/VisSym/EuroVis07/227-234. (Zitiert auf Seite 12)

[SK98] H.-W. Shen, D. L. Kao. A New Line Integral Convolution Algorithm for Visualizing Time-Varying Flow Fields. *IEEE Trans. Vis. Comput. Graph.*, 4(2):98–108, 1998. URL http://dblp.uni-trier.de/db/journals/tvcg/tvcg4.html#ShenK98. (Zitiert auf Seite 11)

[SP00] I. A. Sadarjoen, F. H. Post. Detection, quantification, and tracking of vortices using streamline geometry. *Computers & Graphics*, 24(3):333–341, 2000. URL http://dblp.uni-trier.de/db/journals/cg/cg24.html#SadarjoenP00. (Zitiert auf Seite 13)

[SSKLK13] D. Shreiner, G. Sellers, J. Kessenich, B. Licea-Kane. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.3*. OpenGL. Pearson Education, 2013. URL http://books.google.com.au/books?id=jG4LGmH5RuIC. (Zitiert auf Seite 22)

[STWE07] T. Schafhitzel, E. Tejada, D. Weiskopf, T. Ertl. Point-based Stream Surfaces and Path Surfaces. In *Proceedings of Graphics Interface 2007*, GI '07, S. 289–296. ACM, New York, NY, USA, 2007. doi:10.1145/1268517.1268564. URL http://doi.acm.org/10.1145/1268517.1268564. (Zitiert auf Seite 12)

[Sun03] A. Sundquist. Dynamic Line Integral Convolution for Visualizing Streamline Evolution. *IEEE Trans. Vis. Comput. Graph.*, 9(3):273–282, 2003. URL http://dblp.uni-trier.de/db/journals/tvcg/tvcg9.html#Sundquist03. (Zitiert auf Seite 11)

[TS03] H. Theisel, H.-P. Seidel. Feature Flow Fields. In *Proceedings of the Symposium on Data Visualisation 2003*, VISSYM '03, S. 141–148. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2003. URL http://dl.acm.org/citation.cfm?id=769922.769938. (Zitiert auf Seite 13)

[TWHS04] H. Theisel, T. Weinkauf, H.-C. Hege, H.-P. Seidel. Stream Line and Path Line Oriented Topology for 2D Time-Dependent Vector Fields. In H. Rushmeier, J. J. van Wijk, G. Turk, Herausgeber, *Proc. IEEE Visualization 2004*, S. 321–328. Austin,

U.S.A., 2004. URL http://tinoweinkauf.net/publications/abstheisel04a.html. (Zitiert auf Seite 13)

[TWHS05] H. Theisel, T. Weinkauf, H.-C. Hege, H.-P. Seidel. Topological Methods for 2D Time-Dependent Vector Fields Based on Stream Lines and Path Lines. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):383–394, 2005. URL http://tinoweinkauf.net/publications/abstheisel05a.html. (Zitiert auf Seite 13)

[WEE03] D. Weiskopf, G. Erlebacher, T. Ertl. A Texture-Based Framework for Spacetime-Coherent Visualization of Time-Dependent Vector Fields. In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, VIS '03, S. 15–. IEEE Computer Society, Washington, DC, USA, 2003. doi:10.1109/VISUAL.2003.1250361. URL http://dx.doi.org/10.1109/VISUAL.2003.1250361. (Zitiert auf Seite 12)

[Wij91] J. J. van Wijk. Spot Noise Texture Synthesis for Data Visualization. *SIGGRAPH Comput. Graph.*, 25(4):309–318, 1991. doi:10.1145/127719.122751. URL http://doi.acm.org/10.1145/127719.122751. (Zitiert auf Seite 11)

[Wij02] J. J. van Wijk. Image based flow visualization. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '02, S. 745–754. ACM, New York, NY, USA, 2002. doi:10.1145/566570.566646. URL http://doi.acm.org/10.1145/566570.566646. (Zitiert auf Seite 12)

[WT12] T. Weinkauf, H. Theisel. Flow Visualization and Analysis Using Streak and Time Lines. *Computing in Science & Engineering*, 14(5):78–84, 2012. URL http://tinoweinkauf.net/publications/absweinkauf12c.html. (Zitiert auf Seite 12)

[YWSC12] H. Yu, C. Wang, C.-K. Shene, J. H. Chen. Hierarchical Streamline Bundles. *IEEE Transactions on Visualization and Computer Graphics*, 18(8):1353–1367, 2012. doi:10.1109/TVCG.2011.155. URL http://dx.doi.org/10.1109/TVCG.2011.155. (Zitiert auf Seite 12)

Alle URLs wurden zuletzt am 16.03.2014 geprüft.

Erklärung

Hiermit versichere ich, diese Arbeit selbständig verfasst und nur die angegebenen Quellen benutzt zu haben.

(Sebastian Konle)