

Arbeitsname Bachelorarbeit

VISUALISIERUNG VON STRUKTURVERÄNDERUNGEN IN MOLEKULARDYNA MIKDATEN

Richard Hähne

Geboren am: 10. Februar 1982 in Dresden
Matrikelnummer: 2873574
Immatrikulationsjahr: 2011

BACHELOR WIP

zur Erlangung des akademischen Grades

BACHELOR OF SCIENCE (B.SC.)

Betreuer

Sebastian Grottel

Ludwig Schmutzler

Betreuer Hochschullehrer

Prof. Dr. Stefan Gumhold

Eingereicht am: 31. Dezember 2945

ZUSAMMENFASSUNG

Das vorliegende Dokument befindet sich im grundlegenden Aufbau. Fokus ist eine schnelle Erfassbarkeit bestimmter Inhalte, so dass eher Aufzählungen und Tabellen zu finden sind, die in der Veröffentlichungsfassung gegen Fließtext eingetauscht werden. Teilweise sind auch unformatierte (nicht für Latex formatierte) Abschnitte zu finden.

1. EINLEITUNG

Die visuelle Analyse spielt für komplexe, partikelbasierte Daten, wie sie beispielsweise in der Molekulardynamik entstehen, eine immer wichtiger werdende Rolle. Da die Datensätze immer größer werden, gewinnen abstrahierte Visualisierungen zur Vermittlung eines Überblicks zunehmend an Bedeutung. Kritisch ist hierbei vor allem die Darstellung zeitlicher Entwicklungen. Üblicherweise werden diese als Animation oder in rein abstrakter Form dargestellt. Diese Visualisierungen bieten nur einen schlechten zeitlichen Überblick oder stellen kaum Bezug zum ursprünglich simulierten Ortsraum dar.

Ziel dieser Arbeit ist es daher an einem konkreten Beispiel eine Visualisierung zu entwickeln, die im geometrischen Kontext der Originaldaten die zeitliche Entwicklung der Struktur des Datensatzes darstellt. [... Aufgabenstellung]

2. ANDERE ARBEITEN

2.1. CLUSTERING UND OBERFLÄCHENEXTRAKTION

Atome und Moleküle bestimmten Orten in festen, flüssigen, gasförmigen oder plasmatischen Phasen zuzuordnen, ist eine wesentliche Grundlage der Materialwissenschaft. Durch Diffusion oder Konvektion kann sich die Zugehörigkeit dieser Teilchen zu bestimmten Aggregationen über die Zeit ändern. Aggregationen können Molekülcluster, -tröpfchen und -filamentstrukturen sein.

Für die Zuordnung von Teilchen zu Aggregationen ist das Wissen über deren räumliche Begrenzung Voraussetzung. Dazu ist es notwendig, den Oberflächenverlauf oder Positionsinformationen über das Volumen zu kennen. Die Volumen- und Oberflächenbestimmung von großen Datensätzen ist zum Beispiel bei Skelettextraktionen und Laserscanverfahren untersucht.

2.1.1. SKELETTEXTRAKTION

Skelettextraktionsverfahren nutzen Oberflächen oder Volumen dazu, eine (volumenlose) 1D Skelettkurve zu erstellen, die die Oberfläche oder das Volumen eines 3D Objektes durch ihren Bogenverlauf abbildet und zusätzliche Eigenschaften als Metadaten enthält wie z.B. die Volumendicke [Au+08]. Palágyi vergleicht die Kurve mit einem brennenden Gegenstand, der von allen Seiten gleichmäßig niederbrennt. Die Stellen, an denen sich das Feuer auslöscht, bilden das Skelett des Gegenstandes [Pal08]. Blum nennt sie Medialachsenskelette [Blu67]. Daneben gibt es, abhängig vom Skelettermittlungsverfahren weitere Skelettypen wie z.B. Kurvenskelette [DS06] [CSM07] oder die durch down-sampling davon abgeleiteten Knochenskelette, die vor allem in der Charakteranimation [WPP07] und Meshdeformation [Web+07] eingesetzt werden. Um Skelette aus Oberflächendaten zu berechnen, kommen zum Beispiel Voronoidiagramme zum Einsatz, mit denen der Raum in Regionen aufgeteilt wird und die mediale Oberflächen aus den inneren Kanten und Flächen des Diagramms extrahieren [DS06]. Eine Alternative ist die Nutzung des Reebgraphen, dessen Knoten die kritische Punkte einer auf die Oberfläche angewandten reellwertigen Funktion sind, welche Topologieänderungen der Oberfläche entsprechen [Pas+07]. Der Graph liegt etwa der Laplaceglättung zugrunde, bei der mehrere Vertices eines Meshes zu einem Vertex zusammengefasst werden und so das Objekt verkleinert wird [Au+08]. Weitere Methoden sind das Ausdünnen beim Vorliegen von binären Daten (Daten mit nur zwei Intensitätszuständen, wie z.B. schwarz-weiß Bilder) durch schrittweises Entfernen der äußeren Datenpunkte [Pal08], die Berechnung mittels mittleren Krümmungsflusses, wo die Bewegungsgeschwindigkeit der Normalen eines Punktes auf der Oberfläche vom Krümmungsradius der Oberfläche an diesem Punkt abhängt [Tag+12] oder Meshzerlegungen unter Beachtung der geodätischen Abstände sowie von konvexen Verläufen und Verlinken der Komponenten [KT03]. Die Berechnung von Skeletten aus Volumendaten erfolgt ebenfalls durch Ausdünnung,

etwa von Voxeln durch schrittweises Entfernen der äußeren Voxel [MWL02], durch vorherige Verkleinerung der Volumenmodelle [WL08] oder durch Distanzfeldmethoden, bei denen ein Distanzfeld für jeden inneren Punkt aufgebaut wird, das die Entfernung zum Rand des Objektes enthält und daraus bestimmte Voxel ausgewählt werden, die bei einer Verbindung untereinander eine mediale Oberfläche erzeugen [HF05]. Andere Feldmethoden nutzen einen Potentialwert für innere Punkte durch Einbeziehung mehrerer Randvoxel und sind somit weniger anfällig für Rauschen, jedoch aufwändiger in der Berechnung [Cor+05]. Diese Liste ist nur ein Auszug von üblichen Methoden. Für Kurvenskelette gibt [CSM07] eine umfassende Methodenübersicht.

Den Methoden ist gemeinsam, dass sie die vorhandenen Oberflächen- oder Volumendaten diskretisieren müssen, sollten sie als kontinuierliche Funktionen vorliegen, und die Daten durch eine Bewegung von außen nach innen reduzieren, wodurch eine ursprünglich große Datenmenge durch wenige Knoten repräsentiert werden kann und somit ein Einsatz vor allem in Echtzeitanwendungen findet. Die Verfahren haben dabei mit Problemen wie fehlende Wiedergabe von Oberflächenbeschaffenheiten, insbesondere bei Ausdünnungsverfahren, mit Rauschen oder mit numerischen Instabilitäten bei schlechter Datendiskretisierung zu kämpfen und benötigen teilweise umfangreiche Pre- oder Postprocessingverfahren etwa zum Entfernen von Artefakten oder zum nachträglichen Zentrieren des Skeletts [CSM07].

Die Distanzfeldmethoden bieten Möglichkeiten, den Partikelabstand zum Rand des Clusters zu bestimmen.

Absatz falsch – Für die Bestimmung der Zugehörigkeit von Partikeln zu Clustern können die in der Skelettextraktion verwendeten Methoden jedoch nicht übernommen werden, da sie Fokus auf die Bewahrung der Oberflächentopologie legen, die bei der Erkennung von Strukturereignissen eine geringere Rolle spielt (nicht wahr, da Löcher etc doch interessieren!) und zudem Daten aus der Oberfläche und dem Volumen entfernen und damit die für die Ereigniserkennung notwendige Wanderung von Partikeln nicht verfolgt werden kann. – /Absatz falsch!

2.1.2. AGGLOMERATIONEN

Zusammenhangskomponenten -> Laserscanzeug, Punktwolken

Es existiert eine Implementation für MolCloud für die Zuordnung von Teilchen zu Agglomerationen und die Verfolgung der Wanderung zwischen den Agglomerationen. Die Berechnungen basieren auf zwei Varianten. Eine Variante berücksichtigt nur die geometrische Position der Partikel, eine andere ihre energetischen Eigenschaften (vgl Potentialfeld). Sie liefern jedoch unzureichende Ergebnisse, so werden beispielsweise zu viele oder zu große Cluster erkannt [Gro+07]. Auch besitzt der Benutzer keine Möglichkeit, Einfluss auf die Clustererkennung zu nehmen.

Die vollständige Zuordnung von Partikeln zu Agglomerationen als auch eine benutzergesteuerte Eingabe für die Kontrolle eines Schwellwertes, ab wann Cluster als solche erkannt werden, bieten Konturbäume.

2.1.3. KONTURBÄUME

Konturbäume sind für beliebige Dimensionen geeignet, setzen jedoch wie die Skelettextraktionsverfahren diskrete Wertefelder oder Meshes voraus.

-> nur ganz grob Konturbaum erklären, kommt in den Grundlagen; eher auf die Nutzungsart eingehen! -> die Graphknoten werden wie beim Reebgraph auch aus kritischen Punkten heraus gebildet, vgl zur Skelettextraktion aus Oberflächen mittels Reebgraphen erwähnen! -> hier keine Zeitabhängigkeit, deswegen Algo von 2001 iO -> [Chi+05] geben mit dem Monotonpfadalgorithmus einen Vorschlag für einen schnelleren und vor allem speichersparenden Algorithmus an, indem für die Berechnung der Verbindungs- und Teilungsbäume nur die kritischen Punkte wie Minima, Maxima und Sattelpunkte genutzt werden. Allerdings ist für die Erkennung

der Strukturereignisse jeder Partikel relevant, die in den Zusammenhangskomponenten des Monotonpfadalgorithmus jedoch nicht vollständig vorhanden sind.

Wo verwendet:

the contour tree, which has been used for isosurface extraction [31, 5], for abstract representation of the field [1, 24, 7], to index individual contours [3, 14, 5] and their geometric properties [7], to guide simplification of input meshes [10], and to compare scalar fields [33]. Although recent algorithms for computing the contour tree are efficient, they were originally defined only for simplicial meshes [1, 6], then extended to trilinear meshes [24] and digital images [18]. [CS09, S. 1]

Dateigröße von Datensätzen, Echtzeitdarstellung

2.2. GLYPHDESIGN

TODO

3. GRUNDLAGEN

3.1. KONTURBAUM

Eine Niveaumenge ist die Menge aller Punkte einer Funktion bzw. eines Skalarfeldes, denen ein gleicher Wert zugeordnet ist. Dieser Wert wird nach [CSP10, S. 1] als *Isowert* bezeichnet. Im Zweidimensionalen werden damit *Isolinien* erzeugt, im Dreidimensionalen sind es *Isooberflächen*. Sie finden zum Beispiel bei der Visualisierung von Höhendaten auf topografischen [Hur10] [All+15], von Temperatur-, Wind- und Luftdruckdaten auf meteorologischen Landschaftskarten [Hop96] oder von Gewebestrukturen auf Computertomographieaufnahmen [Tan+14] Verwendung.

Eine *Isooberfläche* kann aus mehreren Zusammenhangskomponenten bestehen, die entsprechend [CSA01a, S. 2] als *Konturen* bezeichnet werden.

Das Skalarfeld wird dabei aus den Abstandswerten der Partikel zum Rand desjenigen Clusters gebildet, in dem sie sich befinden. Daher wird die Bezeichnung *Tiefenwert* für die Werte des Skalarfeldes verwendet. Sämtliche Partikel des Datensatzes sind Teil des Skalarfeldes, auch diejenigen der Gasphase außerhalb von Clustern.

Die *Isooberfläche* schließt ein Volumen ein. Die Partikel innerhalb des Volumens bilden eine Zusammenhangskomponente, eine *Kontur*.

3.1.1. KRITISCHE PUNKTE DER MORSETHEORIE

In der Morsetheorie [Chi+05][28,35] ist ein Punkt dann ein kritischer Punkt, wenn kein Gradient vorhanden ist. Damit setzen die Theoreme voraus, dass sich an diesen Punkten die Topologie der Niveaumenge ändert. Bei einer abschnittsweisen linearen Funktion heißt das, wenn der *Isowert* durch den Punkt verläuft und sich dabei die Topologie der Niveaumenge ändert, dann ist es ein kritischer Punkt. Alle anderen sind reguläre Punkte [CSP10] [Chi+05].

Contour/Isolinie -> Join and Split Tree -> Merge -> Contour Tree [CSA01a, S. 1]

Join: von oben nach unten, Split: von unten nach oben [CSA01b]

Bilder: [Chi+05, S. 1] Fig. 1 [CSP10, S. 44] Fig. 2

3.2. EREIGNISERKENNUNG

Partikel hat ID (durch Arrayindex). Partikel in Zeitschritt 1 & 2 einem Cluster zuordnen und schauen, ob es sich

- im selben, (kann Merge sein, wenn andere Partikel hinzukommen)
- einem anderen schon bei Zeitschritt 1 vorhandenen, (Split)
- einem neuen (Split/Birth)

Cluster befindet. Es können auch auf denselben Clustern mehrere Ereignisse passieren.

3.3. PRÄATTENTIVE WIRKUNG VISUELLER VARIABLEN

- Präattentive Wirkung visueller Variablen (für Priorisierungsreihenfolge visueller Attribute in [Unterabschnitt 5.1.2](#) sowie die Nutzung für Agglomerate). Räumliche Verknüpfungen sind oftmals präattentiv, z.B. Bewegung und (Farbe | Form| 3D-Disparität). Die meisten Verknüpfungen sind nicht präattentiv. => Nutzung möglichst weniger Variablen fördert präattentive Informationsaufnahme => Mehrfachzuweisung (Farbe und Form für Zeit z.B.) vermeiden, auch wenn sie scheinbar zur Verstärkung beitragen.
- falls visuelle Relationen vorkommen: Beispiele von Netzwerken
- falls Zoom eine Rolle spielt: Ansichten, siehe KP InfVis

3.4. GESTALTGESETZE

QUELLEN HINZUFÜGEN, siehe V InfVis, V Gestaltung.

DER EINFACHHEIT HALBER COPYPASTE WIKI, DA NOCH NICHT KLAR, WAS TATSÄCHLICH GEBRAUCHT:

Gesetz der Prägnanz Es werden bevorzugt Gestalten wahrgenommen, die sich von anderen durch ein bestimmtes Merkmal abheben (Prägnanztendenz). Jede Figur wird so wahrgenommen, dass sie in einer möglichst einfachen Struktur resultiert (= „Gute Gestalt“).

Gesetz der Nähe Elemente mit geringen Abständen zueinander werden als zusammengehörig wahrgenommen.

Gesetz der Ähnlichkeit Einander ähnliche Elemente werden eher als zusammengehörig erlebt als einander unähnliche.

Gesetz der Kontinuität Reize, die eine Fortsetzung vorangehender Reize zu sein scheinen, werden als zusammengehörig angesehen.

Gesetz der Geschlossenheit Linien, die eine Fläche umschließen, werden unter sonst gleichen Umständen leichter als eine Einheit aufgefasst als diejenigen, die sich nicht zusammenschließen (D. Katz, Gestaltpsychologie, 1969).

Gesetz der gemeinsamen Bewegung Zwei oder mehrere sich gleichzeitig in eine Richtung bewegende Elemente werden als eine Einheit oder Gestalt wahrgenommen.

Gesetz der fortgesetzt durchgehenden Linie Linien werden immer so gesehen, als folgten sie dem einfachsten Weg. Kreuzen sich zwei Linien, so gehen wir nicht davon aus, dass der Verlauf der Linien an dieser Stelle einen Knick macht, sondern wir sehen zwei gerade durchgehende Linien. Zusätzlich zu diesen von Wertheimer formulierten Gesetzen fand Stephen Palmer in den 1990er Jahren drei weitere Gestaltgesetze.[1]

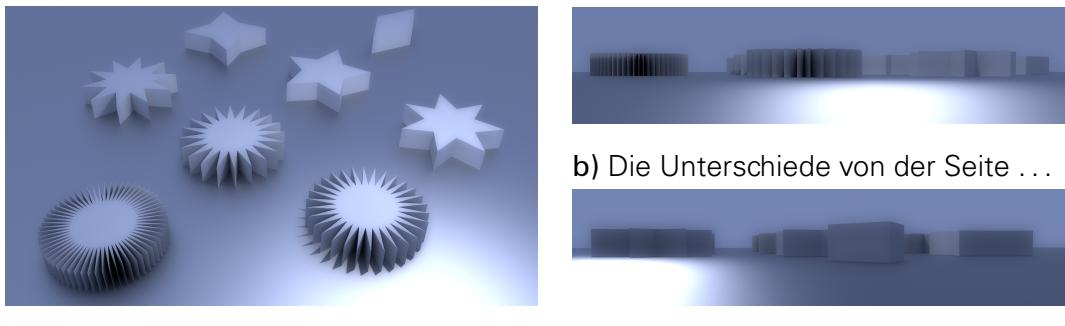
Gesetz der gemeinsamen Region Elemente in abgegrenzten Gebieten werden als zusammengehörig empfunden.

Gesetz der Gleichzeitigkeit Elemente, die sich gleichzeitig verändern, werden als zusammengehörig empfunden.

Gesetz der verbundenen Elemente Verbundene Elemente werden als ein Objekt empfunden.

3.5. FORMGEBUNG

Eine einfache Möglichkeit, unterschiedliche Formen aus einer runden Grundform zu erzeugen, ist ein Zylinder mit nach außen führenden Spitzen wie bei einem Stern, was einem nichtkonvexen Prisma entspricht [[CLM54](#)]. Wie in [Abbildung 3.1](#) zu sehen, sind die Sternformen von vorn und hinten gut zu unterscheiden. Von der Seite und von oben fällt die Abgrenzung voneinander jedoch schwerer. Weiterhin haben die Elemente in der Seitenansicht eine andere Erscheinung

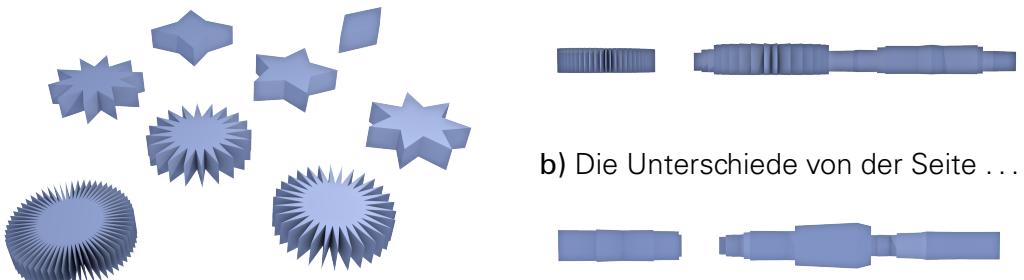


a) Gute Unterscheidung von vorn

b) Die Unterschiede von der Seite ...

c) ... bzw. von oben sind kleiner.

Abbildung 3.1. Formgebung mit sternenförmigem, nichtkonvexen Prisma.
ERKENNBARKEIT SCHLECHTER ALS BEI TRANS UNTEN?



a) Gute Unterscheidung von vorn

b) Die Unterschiede von der Seite ...

c) ... bzw. von oben sind kleiner.

Abbildung 3.2. Formgebung mit sternenförmigem, nichtkonvexen Prisma.
TRANSPARENT

als in der Frontalansicht, so dass dadurch die Verknüpfung zwischen beiden Ansichten mit zusätzlichem kognitiven Aufwand verbunden ist. Deswegen ist diese Art der Formgebung für eine 3D Ansicht nicht geeignet.

Die Anforderung, eine Formgebung zu wählen, die einen ähnlichen Umkugelradius und damit eine einheitliche Größe erhält, einen aus allen Winkeln ähnlichen Durchmesser sowie eine ähnliche Form aufweist, führt zu bestimmten Gruppen von Polyedern. Polyeder sind dreidimensionale Polytope und im engeren Sinne eine Teilmenge des \mathbb{R}^3 , welche ausschließlich durch gerade Flächen begrenzt wird. Darunter zählen unter anderem die fünf *platonischen* [Bog15], die 13 *archimedischen*, die 13 *catalanischen* sowie die 92 *Johnson-Körper* [Joh66]. Eine Charakteristik dieser Polyeder ist ihre konvexe Form, was gleichzeitig die Unterscheidung aus der Distanz erschwert. Die prismatischen Polyeder fallen ebenfalls unter diese Kategorie, werden jedoch aus denselben Gründen ausgeschlossen wie die sternenförmigen Prismen.

Andererseits erfüllen die sternförmigen *Kepler-Poinsot-Körper* („Sternpolyeder“) die Anforderungen. Sie besitzen im Gegensatz zu den vorgenannten Polyedern auch konkave Flächen [Wei15] und können daher, wie in der interaktiven Visualisierung von [Gra15] zu sehen, leicht von den konvexen Polyedern unterschieden werden.

Ebenfalls geeignet ist der Vorschlag, eine Sternform durch sogenanntes „Poken“ zu erzeugen: „By “poke” we mean a function to create a pyramid on each face of a given object. (This is different from Kepler’s stellation operation, which extends the face planes.)“ [Har08, S. 4]

Somit gibt es mit der auf nichtprismatischen Polyedern basierenden Formgebung eine hohe Anzahl verschiedener Formen, die alle die Anforderung an eine einheitliche Größe, an einen aus allen Blickwinkeln ähnlichen Durchmesser und an die Wiedererkennbarkeit gewährleisten. Lediglich die Unterscheidung innerhalb der konvexen Polyeder kann schwerfallen. Am Deutlichsten unterscheiden sich die *platonischen Körper* untereinander sowie die konvexen von den

konkaven, sternförmigen Polyedern.

4. MEGAMOLMODUL MMVIS-STATIC

Die Prämisse beim Entwerfen und Programmieren des Moduls lautete, nah bei den Kernmodulen von MegaMol zu bleiben. Zum Einen erhöht dies die Lesbarkeit für die mit MegaMol Vertrauten und zum Anderen sind die Kernmodule auf Performance.

Dies beinhaltete insbesondere die Verwendung von OpenGL 2.1 im Renderer und Shader, von Zeigern im Callmodul, von der Beschränkung auf Felder im Writer- und Sourcemodul sowie die Verwendung von Bytecode beim Dateiformat selbst.

Der Nachteil der damit verbundenen schlechteren Lesbarkeit des Codes für Neueinsteiger wurde durch das Einfügen zahlreicher Kommentare sowie der Nutzung der glm Bibliothek ausgeglichen, da dadurch die Datentypen in C++ und in GLSL konsistent angelegt werden konnten.

4.1. MEGAMOL MODULSYSTEM

- Callee: Eingänge, angesprochen über ihre Descriptions
- Caller: Ausgänge

Die verfügbaren Module sind je nach Callee/Callertyp unterschiedlich. Beispiele für Typen sind CallGetTransferFunction, MultiParticleDataCall, CallRenderView oder CallRender3D, einstellbar bei der Definition der Callbackfunktion für den Callee/Caller. Diese werden über die Descriptions der Typen angesprochen:

```
1 typedef factories::CallAutoDescription
```

Modifikationen von Eigenschaften von Elementen in anderen Modulen werden über den Callee/Callertyp übermittelt, z.B.

```
1 core::view::CallGetTransferFunction *cgtf = dynamic_cast<core::view::  
    CallGetTransferFunction*>(&call);  
2 // [...]  
3 cgtf->SetTexture(this->texID, this->texSize, this->texFormat);
```

Zugriff auf die einzelnen Partikel einer MMPLD: megamol::core::moldyn::MultiParticleDataCall

4.2. MEGAMOL PARTIKELDATEN

Beim Laden beider mmpld (default und signed distance) kommt der Fehler R6010: abort() has been called, trotz unbegrenzter Speichereinstellung bei MMPLDDatasource. Es wurden

maximal 20 Zeitschritte jeweils geladen (SplitView) bzw. maximal 40 (SingleView) - wegen 32bit (max 1,7GiB RAM Zuweisung wurden angezeigt)! Ein Wechsel auf 64bit hat geholfen (8GiB Arbeitsspeicher noch zu wenig).

4.3. RENDERER

Shader: common.btf für Standardlicht billboard.btf für Billboards in OGL 2.1

4.4. CALL

Wie werden die Variablen im Call gefüllt bzw. die Funktionen zum Füllen aufgerufen? Gefüllt werden sie entweder durch den Calculator oder die Source.

4.5. CALCULATION

4.5.1. DATENGRUNDLAGE

Oft werden Konturbäume bei kontinuierlichen Flächen verwendet (QUELLEN) und es ist eine Diskretisierung dieser etwa mit simplizialen Meshes (QUELLE) notwendig. Darauf kann hier verzichtet werden, da die Eingangsdaten aus Partikeln mit definierten und disjunkten Positionen bestehen und daher bereits in diskreter Form vorliegen.

Weiterhin liegt der *Tiefenwert* für jeden Artikel bereits als vorzeichenbehaftete Gleitkommazahl vor. Dazu werden die Grenzen zwischen der flüssigen Phase und dem Vakuum, bzw. in späteren Zeitschritten der Gasphase bestimmt. Anschließend wird auf dieser Basis eine vorzeichenbehaftete Distanzfunktion aufgestellt, die für alle Partikel die Entfernung zum Rand bestimmt. Somit gibt der *Tiefenwert* die Distanz zum nächsten Partikel an, welches auf dem Rand eines Clusters liegt.

4.5.2. ISOWERT

Der Isowert wird benutzerdefiniert eingestellt.

4.5.3. ZUSAMMENHANGSKOMPONENTE

Der Abstand eines Partikels zum Rand seines Clusters wird mit einem vorzeichenbehafteten *Tiefenwert* gekennzeichnet. Innerhalb von Clustern weisen Partikel eine positive *Tiefe* Tiefenwert auf. Partikel auf dem Rand besitzen den Abstand null, Partikel in der Gasphase, d.h. außerhalb von Clustern, haben einen negativen *Tiefenwert*.

Partikel einer *Kontur* werden über eine radialbasierte Nachbarschaftsbeziehung bestimmt, wobei vom Partikel mit der größten Tiefe, dem sogenannten *Saatpartikel* begonnen wird.

Um diesen Prozess zu beschleunigen, werden beim Auslesen der MMPLD Daten alle Partikel nach ihrem *Tiefenwert* in einer Liste sortiert. Eine speichersplatzsparendere Methode, die auf das Kopieren aller Partikel im Speicher verzichtet, wäre das Nutzen von Zeigern auf den vorhandenen Datensatz, allerdings wäre zum Einen eine Änderung der Daten aufwändig (Lockingmechanismen) und der nächste *Tiefenwert* müsste bei jeder Partikelbehandlung aus dem ursprünglichen MPDC erneut herausgesucht werden.

Ausgehend vom *Saatpartikel* werden die Partikel in der Liste abgelaufen und in eine neue Liste gespeichert. Sollte der nächste Partikel, der den gleichen oder einen etwas geringeren *Tiefenwert* aufweist, in einem bestimmten Radius zum vorhergehenden Partikel liegen, so wird dieser in dieselbe Liste geschrieben, ansonsten in eine andere. Der Radius, mit dem geprüft wird, hängt von der Differenz des *Tiefenwertes* zwischen betrachteten Partikel und

Abbildung 4.1. Datentypen

Minimalpartikel der aktuellen Liste ab. Sollte der Partikel außerhalb dieses Radius' liegen, wird eine neue Liste erstellt.

Eine Ausnahme bilden Partikel mit negativem *Tiefenwert*. Sie werden in eine separate Liste geschrieben und es erfolgt keine Positionsprüfung.

Jede Liste enthält einen Offset. Dort befindet sich der Listenidentifikator. Alternativ eine zusätzliche Liste, die die anderen Listen enthält?

4.5.4. ALGORITHMUS

[CSA01a] [Chi+05, S. 176] (1) Sort all n vertices of the mesh by their function values. (2) Perform a sweep of the n vertices from the smallest function value to the largest function value, and build the join tree. (3) Perform another sweep of the n vertices, now from the largest function value to the smallest function value, and build the split tree. (4) Merge the join tree and split tree together and remove all degree-two nodes in the resulting tree to obtain the contour tree.

4.5.5. DATENBEHANDLUNG

Umwandlung von framebehafteten MMPLD in frameloses MMSE.

Problem: Größe der Daten ist zu umfangreich für den Arbeitsspeicher: Skaliert mit Partikelanzahl und Frames: Bei 3Mio Partikeln 40MB pro Frame. Lösung: Nur die Daten im Speicher halten, die aktuell benötigt werden (bestimmte Anzahl an benachbarten Frames).

MMPLD: Bytecode:

File Header Seek Table Frame [Anzahl im Header]: Anzahl der Partikelliste, Particle List: Header, Partikeldaten

Vertexdaten und Farbdaten für jeden Partikel sind im Speicher sequentiell, direkt hintereinander angeordnet. Weiterhin kann der Radius global oder für jeden Vertex gespeichert sein.[Gro14, S. 3] Der Elementabstand wird durch den Stride bzw. Typ bestimmt (jeweils für den Vertex und die Farben). MMPLD Version 100 ist hier genutzt.

ID = Position des Partikels in der Partikelliste

4.6. DATEIFORMAT

Das [MegaMol Structure Event \(MMSE\)](#) enthält die berechneten Ereignisse in einem Bytecode fest. Der Vorteil im Vergleich zu JSON/XML oder anderen textbasierten Speicherverfahren ist die höhere Performance beim Lesen und Schreiben sowie die kleinere Dateigröße. Die Ereignisdaten sind dabei sequentiell abgelegt

5. VISUALISIERUNGSENTWURF

5.1. TAXONOMIE DER VISUALISIERUNG

Die Berechnung des Clusterverhaltens ergibt eine Datenbank, in der Ereignisse abgelegt sind. Ein Ereignis stellt einen Datensatz dieser Datenbank dar, die Parameter des Ereignisses sind die Datensatzfelder. Die Aufgabe der Visualisierung ist es, den Parametern eines Ereignisses visuelle Attribute zuzuweisen. Diese Zuweisung kann in einer weiteren Datenbank als Relation zwischen Parameter und Attribut festgehalten werden.

Zum leichteren Verständnis erfolgt eine Einteilung der Parameter, der Attribute sowie der Zuweisungen in die beiden in [Abbildung 5.1](#) gezeigten Variablentypen. Der Typ *Zahlenwert* steht dabei für die Nutzung eines kontinuierlichen Zahlenraumes, während der Variablenotyp *Vokabular* eine diskrete, beschränkte Zuordnung mit voneinander disjunkten Elementen beschreibt.

5.1.1. PARAMETER EINESEREIGNISSES

Ein Ereignis besteht aus drei Parametern, die jeweils zwischen ein bis drei Freiheitsgraden aufweisen.

- Zeitpunkt: ein Wert (t)
- Ort: drei Werte (x, y, z)
- Art: ein Vokabular [Split, Merge, Death, Birth]

5.1.2. VISUELLE ATTRIBUTE EINESEREIGNISELEMENTS

Die visuellen Attribute können sowohl als fließender *Zahlenwert* als auch als festes *Vokabular* umgesetzt werden. Beispielsweise bedeutet der Vokabulartyp für das *Positionsattribut* die Festlegung bestimmter Koordinaten für jeden Begriff des Vokabulars.

Jedes Attribut besitzt eine individuelle Anzahl an Freiheitsgraden, die zur Visualisierung genutzt werden können.

- Position: 3 (x, y, z)

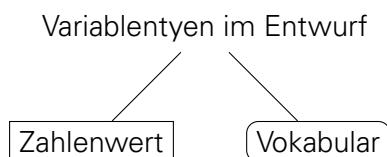


Abbildung 5.1. Taxonomie der Variablenarten

VERGLEICHSBILD MIT WECHSELNDER SÄTTIGUNG UND HELLIGKEIT

Abbildung 5.2. Farbtafel mit unterschiedlicher Sättigung und Helligkeit

- Farbe: 2 (Farbwert/Sättigung, Helligkeit)
- Form: 1, eine Wertzuweisung ist möglich durch die Anzahl der Ecken, Kanten und Flächen. Anderes wie z.B. der Abstand der Ecken vom Körperzentrum kollidiert mit dem Attribut Größe.
- Größe: 1
- Opazität: 1

Da Farbwerte bei geringer Sättigung nicht mehr voneinander unterscheiden werden können, werden diese beiden Farbeigenschaften zusammengefasst. Dadurch kann die Helligkeit leicht als separates Attribut vom Betrachter erkannt werden, wie in [Abbildung 5.2](#) zu sehen.

Bei der *Form* sind mehrere Freiheitsgrade möglich, etwa eine Kombination aus Anzahl der Ecken mit einem variablen Abstand dieser vom Körperzentrum. Der variable Abstand kollidiert jedoch mit dem Attribut *Größe*. Daher wird das Attribut *Form* auf einen Freiheitsgrad beschränkt und es werden die in [Abschnitt 3.5](#) beschriebenen Polyeder verwendet. Da die Anzahl an Polyedern begrenzt ist und die Unterscheidungsmöglichkeit mit zunehmender Flächenanzahl sinkt, ist das Attribut *Form* nur eingeschränkt als Zahlenwert verwendbar.

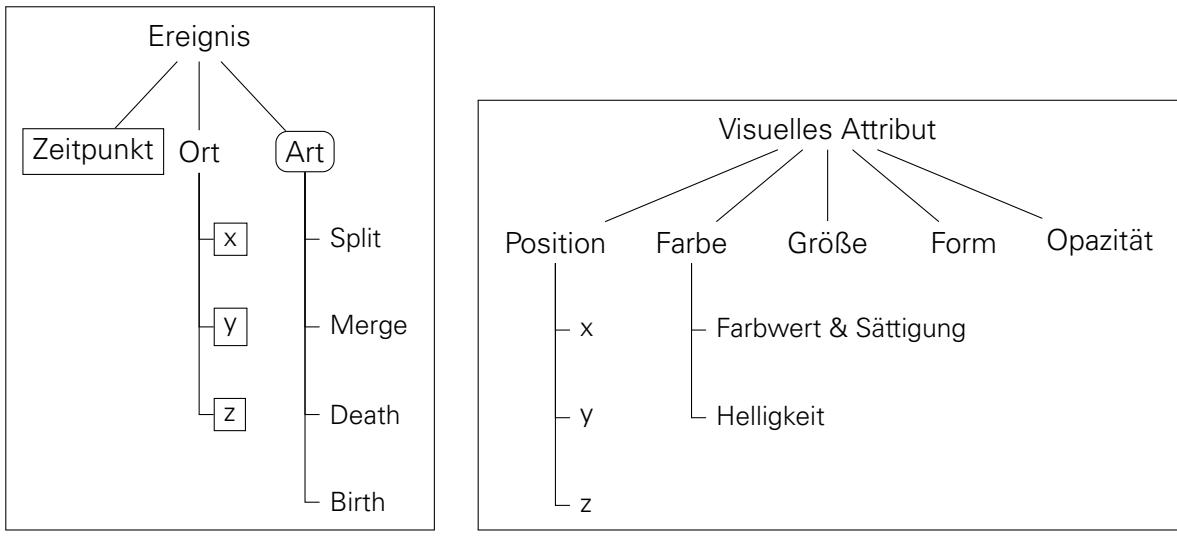
Bei der Bestimmung der Attribute wurde darauf geachtet, ein umfassendes Spektrum der Gestaltgesetze nutzen zu können. Alle visuellen Attribute fallen unter den Einfluss des *Gesetzes der Prägnanz*. Mit der *Position* kann das *Gesetz der Nähe* sowie das *Gesetz der gemeinsamen Region* genutzt werden, eine Kombination aus der *Position* mit den anderen Attributen resultiert in der Verwendbarkeit des *Gesetzes der Kontinuität*. Bis auf die *Position* kann mit allen visuellen Attributen das *Gesetz der Ähnlichkeit* genutzt werden. Das *Gesetz der Geschlossenheit* kann mit dem *Positionsattribut* verarbeitet werden, falls durch entsprechende Zuweisung die Elemente nah beieinander liegen und sie eine geschlossene Form bilden. Mit den hier definierten Attributen kann somit von sechs der zehn in [Abschnitt 3.4](#) aufgezählten Gestaltgesetze Gebrauch gemacht werden.

Texturen als ein eigenständiges visuelles Attribut werden ausgeklammert, da sie sich mit dem Attribut *Farbe* und bei der Nutzung als Oberflächenstruktur mit dem Attribut *Form* überschneiden. Ein Einsatz als Bump Map oder Displacement Map zur Unterstützung des Attributs *Form* wäre denkbar. Aufgrund dessen, dass bereit Polyeder für das *Formatattribut* genutzt werden, wird von einer Verwendung von Texturen allerdings abgesehen.

Animationen bieten weitere visuelle Attribute wie Bewegungsrichtung, Bewegungsbahn, Beschleunigung und Geschwindigkeit als auch die Veränderung der anderen Attribute und eröffnen damit die Nutzung des *Gesetzes der Gleichzeitigkeit* sowie des *Gesetzes der gemeinsamen Bewegung*. Sie werden im Rahmen dieser Arbeit jedoch nicht behandelt.

Die visuellen Attribute lassen sich durch den Betrachter unterschiedlich gut unterscheiden. In der folgenden Sortierung nach ihrer Unterscheidungsgüte stehen die am Leichtesten zu unterscheidenden Attribute ganz oben. (QUELLE)

1. Position
2. Farbwert
3. Größe
4. Form
5. Helligkeit
6. Opazität



a) Ereignis mit Variablentyp

b) Die visuellen Attribute können beide Variablentypen annehmen.

Abbildung 5.3. Taxonomie der Daten und visuellen Attribute

Die Attribute mit höherer Unterscheidungsgüte sollten bevorzugt gewählt werden. Form, Helligkeit und Opazität sind durch die geringere Güte besser als Vokabular- denn als Wertattribut geeignet, da bei den Vokabularen die Anzahl der Begriffe im Vergleich zur Menge der Zahlenwerte gering ist. Dadurch können die Attributseigenschaften weiter auseinanderliegen, was eine leichtere Unterscheidung ermöglicht.

Die

In der *MegaMol* Molekularsimulation wird ausschließlich das visuelle Attribut *Position* zur Darstellung genutzt.

5.1.3. PROBLEM DER LOKALEN UND TEMPORALEN AGGLOMERATION

Die Anzahl der Datensätze, die denselben Ort oder denselben Zeitpunkt aufweisen können, ist unbekannt. Schon eine geringe Anzahl identischer Orte oder Zeitpunkte kann je nach Zuordnung der visuellen Attribute zu einer überlagerten, nicht mehr unterscheidbaren Darstellung führen, wodurch die Erkennung solcher Agglomerate nicht gegeben ist. Daher wird ein visuelles Attribut reserviert, um eine solche Anhäufung von Ereignissen anzudeuten. Die *Häufung* wird bei der Zuweisung zu den visuellen Attributen wie ein Ereignisparameter behandelt. Abhängig von dieser Zuweisung in Abschnitt 5.2 können zwei Werte für die lokale und temporale Häufigkeit notwendig sein. Dieser wird jedem Ereignis zugewiesen und wird beim Auffinden von identischen Ortsparametern bzw. Zeitparametern in den Datensätzen inkrementiert.

Die Begriffe *Agglomeration* und *Häufung* sind hier synonym.

Bei der Visualisierung von Agglomeraten kann die natürliche Wirkung (QUELLE) der visuellen Attribute genutzt werden. Groß, opaque und dunkel stehen für eine starke Agglomeration. Klein, transparent und hell für eine niedrige.

5.1.4. VERBUNDENHEIT

Wenn sich ein Cluster teilt (Split) und sich die resultierenden Cluster wiederum teilen oder mit anderen Clustern verschmelzen (Merge), so kann man diesen Ereignissen Vorgänger (Eltern) sowie Nachfolger (Kinder) zuordnen. IST DAS RICHTIG?

Gesetz der Geschlossenheit Gesetz der fortgesetzt durchgehenden Linie Gesetz der verbundenen Elemente

Tabelle 5.1. Zuweisungsmöglichkeiten der Parameter zu den visuellen Attributen bei Kopplung des *Ortes* an die *Position*. Zur Anzeige der Häufung H wird die *Opazität* reserviert. Die Zeit wird in einigen Varianten nicht kodiert.

Attribut	Position x	Position y	Position z	Farbwert & Sättigung	Helligkeit	Eckenanzahl	Größe	Opazität
Ort x	Ort y	Ort z	Art	Zeit	Zeit	Zeit	Zeit	H
	Ort y	Ort z	Art					H
	Ort y	Ort z	Art					H
	Ort y	Ort z	Art					H
Ort x	Ort y	Ort z	Zeit	Art	Zeit	Zeit	Zeit	H
	Ort y	Ort z	Zeit	Art				H
	Ort y	Ort z	Zeit	Art				H
	Ort y	Ort z	Zeit	Art				H
Ort x	Ort y	Ort z	Zeit	Zeit	Art	Zeit	Zeit	H
	Ort y	Ort z	Zeit	Zeit	Art			H
	Ort y	Ort z	Zeit	Zeit	Art			H
	Ort y	Ort z	Zeit	Zeit	Art			H
Ort x	Ort y	Ort z	Zeit	Zeit	Art	Zeit	Zeit	H
	Ort y	Ort z	Zeit	Zeit	Art			H
	Ort y	Ort z	Zeit	Zeit	Art			H
	Ort y	Ort z	Zeit	Zeit	Art			H
Ort x	Ort y	Ort z	Zeit	Zeit	Art	Zeit	Zeit	H
	Ort y	Ort z	Zeit	Zeit	Art			H
	Ort y	Ort z	Zeit	Zeit	Art			H
	Ort y	Ort z	Zeit	Zeit	Art			H

5.2. PARAMETER-ATTRIBUTSUWEISUNG

In der *MegaMol* Simulation wird der Ort des Ereignisses mithilfe der Teilchenposition dynamisch visualisiert. Insofern kann angenommen werden, dass für eine nahtlose Benutzbarkeit zwischen der *MegaMol* Animation und der statischen Visualisierung die Zuweisung des Elementattributes *Position* an den Parameter *Ort* empfehlenswert ist. Weiterhin ist die Reservierung der *Opazität* für die *Häufung* die einfachste Möglichkeit, denn dies hat zwei Vorteile. Zum Einen ist keine nachträgliche Modifizierung der visuellen Attribute der einzelnen Ereignisse notwendig, denn die Opazitätseigenschaft sorgt durch Überlagerung für eine von der Anzahl direkt abhängige Hintergrundverdeckung. Dadurch werden zum Anderen auch keine Informationen über die Häufigkeit im Datensatz benötigt. Hingegen muss zum Beispiel eine Größenveränderung nachträglich für jedes Ereignis im Agglomerat auf der Basis eines temporalen oder lokalen *Häufungswert* durchgeführt werden.

Darüberhinaus wird jedem Ereignisparameter nur eine visuelle Attribut zugeordnet, denn Mehrfachzuweisungen sollten wegen einer Verminderung der präattentiven Wahrnehmung (VERWEIS AUF GRUNDLAGEN) vermieden werden.

Daraus ergeben sich die in [Tabelle 5.1](#) aufgeführten Visualisierungsmöglichkeiten.

Derzeit ist nicht bekannt, wie groß die örtliche Konzentration von Ereignissen ist. Sollte sie eine kritische Anzahl überschreiten und das der *Häufung* zugewiesene Attribut keine sinnvollen Ergebnisse mehr liefern können, so kann der Ort des Ereignisses einem anderen Attribut zugewiesen werden. Weiterhin kann das Loslösen des *Ortes* von der Position sinnvoll sein, wenn für eine Auswertung der Ort des Ereignisses nur eine untergeordnete Rolle spielt. Vorschläge für entsprechende Zuweisungsmöglichkeiten sind in [Tabelle 5.2](#) zu finden.

Da jedoch in der Molekularsimulation der Ort eines Ereignisses der visuellen Position zu-

Tabelle 5.2. Ausgewählte Zuweisungsmöglichkeiten der Parameter zu den visuellen Attributen bei Loslösung des *Ortes* von der *Position* sowie Trennung der *Häufung H* von der *Opazität*. Nicht jeder Parameter wird kodiert. UNVOLLSTÄNDIG

Attribut	Position x	Position y	Position z	Farbwert & Sättigung	Helligkeit	Eckenanzahl	Größe	Opazität
Zeit	H			Art				
Zeit	Art			H	Ort x			
Ort x	Art							H
Ort x	Zeit						H	

gewiesen ist und die Visualisierung in der vorliegenden, ersten Iteration mit der Simulation harmonieren soll, erfolgt die Zuweisung des Ortes ausschließlich an die *Position*, so dass die weitere Betrachtung auf die in [Tabelle 5.1](#) gezeigten Zuweisungsmöglichkeiten beschränkt wird.

5.3. MOCKUPS

5.3.1. WERKZEUG

Zur Überprüfung einer sinnvollen Zuweisung der Parameter zu den Attributen dienen Mockups. Aufgrund der Vielzahl der in [Abschnitt 5.2](#) genannten Kombinationsmöglichkeiten wird eine programmatische Erstellung der Mockups gewählt. Da eine schnelle Umsetzbarkeit im Vordergrund steht, wird *Unity3D* als Entwicklungsumgebung gewählt. Ein Vorteil dieser Engine ist die Möglichkeit, rasch Prototypen erstellen zu können sowie die einfache Umsetzbarkeit für verschiedenste Plattformen, ohne auf Containermanager wie [Docker](#) angewiesen zu sein. Letzterer Umstand ist für den möglichen Einsatz der Mockups als Evaluationswerkzeug von Bedeutung. Die Performance spielt beim Mockup eine untergeordnete Rolle.

Beim Deploy der *Unity3D* WebGL Anwendung kam es auf bestimmten Apachekonfigurationen zu Problemen. Dies war durch eine [manuelle Umbenennung von Dateien und Verweisen](#) in der durch *Unity3D* generierten Anwendung zu lösen. Von offizieller Seite wurde dies als [Bug markiert](#). SOLCHE SÄTZE HABEN IN DER BA VERMUTLICH NIX ZU SUCHEN?

5.3.2. DATENSATZ

Um eine Vergleichbarkeit der verschiedenen Mockups zu gewährleisten, liegt ein fester Datensatz in [Tabelle A.1](#) zugrunde. Sämtliche Werte sind einheitenlos und wurden für die Parameter *Ort* und *Zeit* so gewählt, dass eine Überlappung der Ereigniselemente von nebeneinanderliegenden Ereignissen bei einer Zuweisung an das visuelle Attribut *Position* nicht auftritt, da andernfalls durch das *Gesetz der verbundenen Elemente* ein falsches Abbild der zugrundeliegenden Daten übertragen werden würde. Der Zahlenbereich für die Zeit ist auf das Intervall [1, 22] begrenzt. Da sich in der Simulation die meisten Elemente mit zunehmender Zeit vom Zentrum entfernen, wird die x-Koordinate des Ortes mit [Gleichung 5.1](#) direkt an die Zeit t gekoppelt.

$$x = \left\lfloor \frac{4}{3} \cdot t + 0,5 \right\rfloor \quad (5.1)$$

Aufgrund des zylinderförmigen Raumes in der Simulation wird sowohl für die y-Koordinate als auch für die z-Koordinate des Ortes ein identisches und kleineres Intervall von [1, 10] gewählt.

Die Zuweisung der *Art* des Ereignisses erfolgt manuell. Da die simulierten Partikel in der *MegaMol* zu Beginn der Simulation einen einzigen Cluster in flüssiger Phase bilden, kommen bei kleinen Zeitwerten vor allem Splits vor.

Zur Darstellung der in [Unterabschnitt 5.1.3](#) beschriebenen Häufung, werden die Ereignisse 6 – 9, 35/36 sowie 46 – 48 jeweils gruppiert und händisch an denselben Ort und dieselbe Zeit gesetzt.

5.3.3. ZUWEISUNG DER ZAHLENWERTE

Zur Gewährleistung der in [Unterabschnitt 5.3.2](#) beschriebenen Vermeidung unnötiger Überlappungen der Ereigniselemente wird der Standardwert des Attributs *Größe*, d.h. wenn das Attribut keinem Parameter zugewiesen ist, entsprechend skaliert. Um nah bei der Visualisierung der Molekularsimulation selbst zu bleiben, wird in den Mockups, die nicht das Attribut *Form* beinhalten, eine Kugel verwendet.

Die Attribute in den Mockups weisen folgende Zahlenbereiche auf:

- x-, y-, z-Position: $[-\infty, \infty]$, beim Parameter *Ort* erfolgt eine direkte Zuweisung
- Farbwert: $[0, 340]^\circ$ bei 100% Sättigung ([Hue-Saturation-Brightness \(HSB\)](#) Modell)
- Helligkeit $[40, 100]\%$, Standardwert 100% ([HSB](#)-Modell)
- Form: besitzt keinen Zahlenbereich, sondern beinhaltet verschiedene Formen von Polyedern sowie die Kugel als Standardwert
- Skalierung: $[0.25, 2]$, Standardwert 1
- Opazität: $[40, 100]\%$, Standardwert 100%

Die Mindesthelligkeit b_{\min} von 40% dient dazu, um noch eine Farbwertunterscheidung treffen zu können, genauso wie die Mindestopazität von ebenfalls 40%. Der Maximalfarbwert h_{\max} von 340° wird festgelegt, um eine Unterscheidung zu den niedrigen Farbwerten zu erhalten.

Da die Grenzwerte der visuellen Attribute a_{\min} und a_{\max} aus [Abschnitt 5.3](#) bekannt sind und sich für die Parameter p aus [Tabelle A.1](#) ein minimaler und maximaler Wert p_{\min} und p_{\max} bestimmen lässt, wird für die Berechnung der Attributwerte die Linearfunktion in [Gleichung 5.2](#) verwendet.

$$\begin{aligned}
 a &= m \cdot p + n \\
 a_{\max} &= m \cdot p_{\max} + n \\
 a_{\min} &= m \cdot p_{\min} + n \\
 m &= \boxed{\frac{a_{\max} - n}{p_{\max}}} \\
 n &= a_{\min} - m \cdot p_{\min} = a_{\min} - \frac{a_{\max} - n}{p_{\max}} \cdot p_{\min} \\
 &= a_{\min} - a_{\max} \cdot \frac{p_{\min}}{p_{\max}} + n \cdot \frac{p_{\min}}{p_{\max}} \\
 n \cdot p_{\max} &= a_{\min} \cdot p_{\max} - a_{\max} \cdot p_{\min} + n \cdot p_{\min} \\
 n \cdot (p_{\max} - p_{\min}) &= a_{\min} \cdot p_{\max} - a_{\max} \cdot p_{\min} \\
 n &= \boxed{\frac{a_{\min} \cdot p_{\max} - a_{\max} \cdot p_{\min}}{p_{\max} - p_{\min}}}
 \end{aligned} \tag{5.2}$$

Anschließend erfolgt eine Rundung zur handlichen Verwendung der Ergebnisse.

$$a = \lfloor m \cdot p + n + 0,5 \rfloor \tag{5.3}$$

Die Berechnungsergebnisse sind in [Abschnitt A.1](#) zu finden.

Tabelle 5.3. Festlegung der Attributeigenschaften für den Parameter *Art* in den Mockups.

Art	Farbwert [°]	Helligkeit [%]	Position x	Form	Größe
Death	0	40	1	T	0,25
Merge	55	60	11	H	0,83
Split	245	80	21	O	1,42
Birth	145	100	31	D	2

5.3.4. ZUWEISUNG DES VOKABULARS

Da die Mächtigkeit M des Vokabulars von *Art* bekannt ist, kann das Intervall v des Attributs a in [Gleichung 5.4](#) zur rechnerischen Ermittlung der Attributwerte a_i genutzt werden.

$$\begin{aligned} M &= 4 \\ \Rightarrow p_{0...3} &= \left(0, \frac{1}{3}, \frac{2}{3}, 1\right) \\ v &= a_{\max} - a_{\min} \end{aligned} \tag{5.4}$$

$$a_i = \{p_i \cdot v + a_{\min} \mid 0 \leq i \leq M - 1\}$$

Ausnahmen bilden die Attribute *Farbe* und *Form*. Wegen der kleinen Mächtigkeit des Vokabulars und zur Sicherung einer optimale Aufteilung des Farbwerts, wird dieser in [Tabelle 5.3](#) manuell definiert. Bei dem Attribut *Form* werden zur optimalen Unterscheidung vier Varianten aus der Menge der *platonischen Körper* ausgewählt.

5.3.5. AUSNAHME: KONSTANTER ATTRIBUTSWERT BEI OPAZITÄT

Aus den in [Abschnitt 5.2](#) aufgeführten Gründen ist es sinnvoll, die *Opazität* der *Häufung* zuweisen und sie mit einem konstanten Variablentyp auszustatten, so dass keine zusätzliche Ereigniswerte für die *Agglomeration* benötigt werden. Dieser konstante Wert wird auf 50% Opazität festgesetzt.

5.3.6. UNTERSCHIEDBARKEIT VON POLYEDERN

Da die Unterscheidung von Polyedern mit zunehmender Flächenanzahl schwieriger wird, findet im Mockup eine Begrenzung auf die fünf *platonischen Körper* mit einer geringen Ecken- und Flächenanzahl statt.

- T** Tetraeder mit 4 Ecken und 4 Flächen
- O** Oktaeder mit 6 Ecken und 8 Flächen
- H** Hexaeder mit 8 Ecken und 6 Flächen
- I** Icosaeder mit 12 Ecken und 20 Flächen
- D** Dodekaeder mit 20 Ecken und 12 Flächen

Die Abkürzungen folgen dem Vorschlag von Baierl [[Bai04](#), S. 42]. Bei der Modellierung in *Blender* wurde für die Meshes die maximale Dimension von 1 (einheitenlos) entsprechend der Dimension der kugelförmigen Standardelemente in *Unity3D* gewählt.

5.3.7. VERBUNDENHEIT

Visuelle Relationen. Falls überhaupt möglich/benötigt (erstmal Ereignisse erstellen in *MegaMol*). Linien bieten sich an (Splines/Nurbs mal gucken).

NOCH NICHT IMPLEMENTIERT. WEITEREN DATENSATZ ERSTELLEN MIT ID'S AUS Tabelle A.1 mit der Struktur

```
1 /* MySQL */
2 CREATE TABLE relations (
3     ID int NOT NULL AUTO_INCREMENT, /* int reicht fürs Mockup. */
4     SourceID int NOT NULL,
5     TargetID text NOT NULL, /* String , da unbekannte Anzahl an Targets. */
6     PRIMARY KEY (ID)
7 )
```

Alternativ Kinder/Eltern IDs für jedes Ereignis speichern.

6. ENTWURFSAUSWERTUNG

In den Abbildungen ?? sind Beispiele dafür zu sehen, wenn die *Häufung* anderen Attributen außer der *Opazität* zugewiesen wird.

6.1. GLYPHEN

- Polygone nicht so gut geeignet, da Unterscheidung auf Entfernung schwer möglich und auch hier noch Abhängigkeit vom Blickwinkel - daher zweidimensionale, vordefinierte Bilder an Eventtypen gekoppelt, immer zur Kamera ausgerichtet vgl. Sprites

Abbildung 6.1. Mockups mit freier Positionszuweisung.

7. NOTIZEN GROTEL

7.1. VIS07GROTEL

[Gro+07]: Ideen für Clustering-Kriterien aus denen man vielleicht die Zusammenhangs- und Gas-Erkennung ableiten kann.

Qualität der Clustererkennung: Monomer = point based sphere; Cluster = Ellipse; flow group = Pfeil, Linie (graph) flow groups: Fluss von Molekülen zwischen zwei Clustern. Quantitativ: Zusätzlicher schematischer GraphView (zur 3D-Partikelanzeige). Synchronisierter Filter/Selektion zwischen den Views. Dadurch bekommt der Nutzer Informationen über die erkannten Cluster, deren zeitliche Entwicklung und ihre Interaktion untereinander und mit den umgebenden Monomeren.

Clustererkennung: Zwei Methoden.[Gro+07, S. 6] Die Energetische ist genauer, die geometrische erkennt zu viele und zu große.

- Geometrische, vier Nachbarpartikel.
- Energetische: defines two molecules as clustered if the sum of their potential energy and their relative kinetic energy is negative[12]. T. L. Hill. Molecular clusters in imperfect gases. The Journal of Chemical Physics, 23:617–622, April 1955.

-> which they [] refer to as merge-split graph, visualizing important events in the features' evolution like births, merges, splits and death: as we do: [] D. Laney, P.-T. Bremer, A. Mascarenhas, P. Miller, and V. Pascucci. Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities. IEEE Trans. Vis. Comput. Graph., 12(5):1053–1060, 2006.

Datenverwaltung (big data):

- The molecule positions can be stored directly, or they can be placed in a positional hierarchy using relative quantized coordinates as presented in [14].
- we implicitly encode cluster membership through the hierarchy, representing clusters by inner nodes: Eine Gruppe pro Cluster?
Molecular clusters are identified by a natural number. The value zero is used to represent molecules that are not part of a cluster.
- Daten werden zwischen zwei Konfigurationen/Zeitschritten interpoliert, um eine flüssige Darstellung zu ermöglichen
- Zyklische Randbedingung heißt, Partikel treten an der Simulationsgrenze oben aus und unten wieder ein?

7.2. TOPOINVIS09GROTEL

[Gro+09]: Extraktion von Strukturen. Der Ansatz ist totaler Schwachsinn, aber vielleicht kannst Du aus den Klassifikations-Ideen etwas ableiten und die Struktur in lineare Segmente und Knotenpunkte zu zerlegen.

We propose two different approaches to classify nodes in the graph to be junctions or not: based on the degree of the node or based on the positions of the direct neighboring nodes

8. NOTIZEN CONTOUR TREE

8.1. BAJAJ1997CONTOURSPECTRUM

Contour Spectrum als Benutzerinterface für die Echtzeitquantifizierung in der Visualisierung von Isokonturen. Ist eine Signatur über Skalardaten und Konturattributen. Nutzung von Oberflächen, Volumen und Gradienten, präsentiert als Signaturgraphen (Ausdrucke der Skalardaten). Zeitabhängig: Ausdrucke über Zeit mit 2D Interface. Nutzer kann Isowert und Zeitschritt einstellen.

8.2. CARR2001COMPUTINGCOUNTOURTREES

[CSA01a]

Isolines = Contours. „Isolines, often called contours, are the curves consisting of points at a given height that can be seen on any topographic map.“ [CSA01a, S. 1]

Niveaumenge besteht aus Punkten p , die bei Anwendung einer Funktion f dieselbe Höhe h aufweisen $f(p_i) = h_i$. In 2D sind es Isolinien, in 3D Isooberflächen [CSA01a, S. 2]. Hier wird generell von Konturen gesprochen.

Bei Betrachtung von h als Zeit und der Beobachtung der Veränderung der Niveaumengen über die Zeit kann das Entstehen, das Teilen, die Geschlechtsänderung, das Verbinden und das Verschwinden beobachtet werden. Der Konturbaum ist ein Graph, der die Konturen der Niveaumengen im Verbinden und Erscheinen oder Trennen und Sterben verfolgt.

[CSA01a, S. 3] Octrees, span space, interval trees können keine Konturen unterscheiden, weil sie jede Überschneidung einer Zelle und einer Kontur als separates Object behandeln. Im Gegensatz dazu nutzen Extrema graph, segment tree und contour tree die Verbindung der einzelnen Konturen. Ausgehend von einer Saatzelle (Startzelle, die eine Kontur schneidet), kann man die danebenliegenden Zellen ablaufen, bis sich die Kontur schließt. Die Saatzellen können unter anderem mit Konturbäumen bestimmt werden.

8.2.1. MORSETHEORIE

Kritische Punkte, an denen sich die Niveaumenge ändert. Nicht alle Punkte wichtig. Z.B. Geschlechtsänderungen sind uninteressant. Folgende vier kritische Punkte interessieren:

8.2.2. KONTURBAUM

ϵ -Nachbar schneidet mindestens zwei Konturen bei $f(x) + \delta = \text{Join}$ $f(x) - \delta = \text{Split}$

ϵ -Nachbar schneidet keine Konturen bei $f(x) + \delta = \text{lokales Maximum}$ $f(x) - \delta = \text{lokales Minimum}$
Konturen sind äquivalent, d.h. sie gehören einer Konturklasse an, wenn sie:

- nicht einen kritischen Punkt passieren und
- sie zur selben zusammengehörigen Komponente gehören und
- kein Join über der Minimalhöhe im Höhenbereich der Konturen liegt und
- kein Split unter der Maximalhöhe im Höhenbereich der Konturen liegt

= offenes Intervall = unendliche Konturklasse.

Konturen, die einen kritischen Punkt passieren, sind das einzige Mitglied ihrer Konturklasse = endliche Konturklasse.

Konturbaum: Knoten = Superknoten, Kante = Superbogen

Innerer Superknoten kritischer Punkt, an dem Erzeugung und Zerstörung jeweils einer oder mehrerer unendlicher Konturklassen stattfindet

Blattsuperknoten lokales Maximum, an dem eine unendliche Konturklasse erzeugt oder lokales Minimum, an dem eine unendliche Konturklasse zerstört wird

Superbogen eine unendliche Konturklasse, die am Quellknoten erzeugt und am Zielknoten zerstört wird

Konturbaum wird um nicht kritische Punkte erweitert, indem diese in die zugehörige Konturklasse aufgenommen werden. Die Darstellung erfolgt am jeweiligen Bogen der Klasse als kleine Knoten = erweiterter Konturbaum.

8.2.3. ALGORITHMUS

Zwei Schritte:

1. Join und Split Tree zur Erkennung von Joins und Splits erstellen
2. Bäume zusammenfügen zum Konturbaum

VERBINDUNGS- UND TEILUNGSBÄUME

Verbindungsbaum beinhaltet alle Verbindungen im Konturbaum. Ist dual zum Teilungsbaum (enthält alle Teilungen) durch Negieren der Höhen.

KONTURBAUM

up-arc Bogen nach oben

down-arc Bogen nach unten

up-degree Bogenanzahl nach oben

down-degree Bogenanzahl nach unten

upper-leaves up-degree = 0

lower-leaves down-degree = 0

Reduktion der Knoten in den Verbindungs- und Teilungsbäumen mit up- und down-degree = 1.

Algorithmus für große Datensätze wird erwähnt, nicht gefunden: http://www.comp.leeds.ac.uk/scshca/journal_papers.html

8.3. CARR2009 REPRESENTING INTERPOLANT TOPOLOGY

Framework zur Erfassung aller signifikanten topologischen Eigenschaften der Eingabedaten. Theoretische Grundlage für jedes zufällige Mesh, Kachelung (Tessellation) oder Verbindungsregel. Methoden vereinfacht für die Berechnung von Konturbäumen von Bezierinterpolierungen, Marching Cubes [16, 19], Marching Hypercubes [2] and digital image connectivity.

Keine Interpolierende ist optimal für alle Eingabedaten oder Anwendungsfälle, die Interpolierende sollte wählbar sein.

Endlicher Automat, der die Topologie bei einer gegebenen Interpolierenden verfolgt und so die Entwicklung der Isolinie in einer einzelnen Zelle verständlicher macht.

8.4. CARR2010 FLEXIBLE ISOSURFACES

Nutzung des Konturbaums zur Visualisierung nutzerangepasster Konturen mit Isowerten.

Isowert. Der Passende ist nicht immer bekannt. Isowert muss durch Algorithmus oder durch den Nutzer bestimmt werden. Nutzer muss die Daten erkunden, um einen passenden Isowert zu finden: Trial & Error.

[CSP10, S. 43] Konturbaum ermöglicht Verfolgung der Konturen durch eine flexible Isooberfläche (viele Konturen). Benutzerschnittstellen: 1) direkte Kontrolle der Konturen und Isowerte 2) indirekt über den Konturbaum

Ein passiver Konturbaum (nur Visualisierung) wird umgewandelt zu einer aktiven Manipulation einzelner Konturen.

Annahme 1: Nicht alle Konturen sind wichtig. Rauschen und viele Konturen führen zu einem riesigen und unübersichtlichen Konturbaum. Reduktion nötig.

8.4.1. MORSE

Kritische Punkte: Maxima, Sattelpunkte (Gradient der Funktion = 0), Rest ist regulärer Punkt. Sattelpunkte weisen auf Änderungen bei den Zusammenhangskomponenten oder beim Konturgeschlecht hin. Die Morsetheorie nimmt an, dass alle kritischen Punkte unterschiedliche Isowerte haben und es keine flachen Regionen haben (d.h. regions of measure greater than zero with 0 gradient -was heißt das?).

Morsetheorie über den Reebgraphen interessant: Ein Graph über topologische Beziehungen zwischen Konturen, die über das Zusammenziehen jeder Kontur zu einem einzigen Punkt erzeugt werden, während der Zusammenhang zwischen den Konturen erhalten bleibt. D.h. jeder zusammenhängender Pfad wird zu einem zusammenhängenden Pfad im Reeb Graphen zusammengezogen. Dieser Graph wird Konturgraph genannt[2]: Kanten = Superbögen, Knoten = Superknoten.

[2] R.L. Boyell, H. Ruston, Hybrid techniques for real-time radar simulation, in: Proceedings of the 1963 Fall Joint Computer Conference, IEEE, 1963, pp. 445–458.

Alle Superknoten = kritische Punkte, aber nicht umgekehrt! An den kritischen Punkten, an denen sich die Zusammenhangskomponente nicht ändert, brauchen keine Superknoten zu sein. Daher: connectivity critical points = critical points at which connectivity changes connectivity regular points = all other critical and regular points

Bild 2 auf [CSP10, S. 44] ist super zur Visualisierung!

Diese 1:1 Verbindung zwischen Konturbaum und den Konturen des zugrundeliegenden Skalarfeldes ist die Basis für die flexible Isooberfläche.

Konvention 1: Eine Kontur c ist eine einzelne Zusammenhangskomponente einer Niveaumenge bei Isowert h und kann durch das Tupel $c = (s, h)$ symbolisiert werden, mit s als Superbogen. Die Kontur gehört also zum Superbogen: $c \subset s$.

Am Superknoten kann die Kontur durch mehrere Tupel angegeben sein, eines für jeden anliegenden Superbogen. Das ist allerdings redundant, da die Kontur ja nur am Superbogen beim gegebenen Isowert existiert und Konturen nie am Superknoten extrahiert werden. Diese Annahme ist eine Form einer symbolischen Störung und ist allen Isooberflächenextraktionsmethoden gemeinsam, wie etwa Marching Cubes, wo die Vertex über oder unter der Isooberfläche, aber nie genau auf dieser Isooberfläche klassifiziert werden. [22]

[22] W.E. Lorensen, H.E. Cline, Marching cubes: A high resolution 3D surface construction algorithm, Computer Graphics 21 (4) (1987) 163–169.

1:1 impliziert hilfreiche Eigenschaft:

Eigenschaft 1: Ein monotoner Pfad im Skalarfeld passt auf einen monotonen Pfad im Konturbaum. Jeder monotone Pfad im Konturbaum ist das Abbild von mindestens einem monotonem Pfad im Skalarfeld.

8.4.2. VORANGEGANGENE ARBEITEN

[CSP10, S. 45]

Isooberflächenextraktion: Polyhedral mech: - Marching cubes [22] - continuation [44]: Saat, für jede Kontur: da einzelne Konturen interessant, wird diese Methode gewählt. [44] G. Wyvill, C. McPheeers, B. Wyvill, Data structure for soft objects, Visual Computer 2 (1986) 227–234.

Konturbaum: Für - Konturindizierung [2,4,42] - Terrainbeschreibung [11,20,35,37] - Volumendaten [1,33] - Featureddetection [39,45] - Isosurfaceextraktion [4,42] - Datenvereinfachung [6,8] - Transferfunktionsdesign [39] - Extraktion von Kontureigenschaften [6,17].

Algorithmen: Takahashi et al. [37, 39], van Kreveld et al. [42], Carr, Snoeyink and Axen [5], Pascucci [28] and Cole-McLaughlin [30] (Betti numbers, um das topologische Geschlecht der Konturen zu bestimmen), Chiang et al. [7].

Weitere Themen, hier unrelevant: Lokale Oberflächenwahl, Isooberflächenauswahl, Transferfunktionsdesign.

8.4.3. PFADQUELLEN FÜR EINZELKONTUREXTRAKTION

- minimal seed sets [42]: Vier Nachteile - extrema graph [15]: zuviele Saatzellen

Pfadverfolgung nur bedingt interessant, da nebeneinanderliegende Partikel ähnliche Tiefen aufweisen werden. Daher werden evtl wenig Senken/Minima, Sätteln oder Maxima auftauchen bzw. die Saat ist durch die minimale Tiefe schon vorgegeben und es erfolgt eine Verfolgung nach der Tiefe und nicht des Partikelortes.

Join und Split nach Union Find [40]. Kanten werden zur union-find Struktur hinzugefügt = Join-Split. Bögen beider Bäume zusammenfügen, überlaufende Bögen werden rausgeschmissen.

8.4.4. VEREINFACHUNG: LOKALE GEOMETRISCHE GRÖSSEN

Topologische Vereinfachung des Konturbaums/des Datensatzes.

Lokale geometrische Größen der Konturen. Z.B. Kontur/Vertex ist ober- oder unterhalb eines Sattelpunktes.

[CSP10, S. 53] Evolving a contour: Hochlaufen und Runterlaufen entlang des Baumes an den kritischen Punkten.

8.4.5. ERGEBNIS

Further work is required on the best geometric measures for tree simplification, on the use of local geometric measures for automatic feature recognition, on improved methods for displaying the contour tree visually, and on methods for simultaneous simplification of the contour tree and the contours extracted. [CSP10, S. 56]

8.5. CHIANG2005CONTOURTREESUSINGMONOTONEPATHS

[Chi+05]

Visualisierung von Konturen: Nur einen Teil der Daten sortieren, die für die Ausgabe interessant sind.

Räumliche Daten: electromagnetic waves in nuclear spin tomography, heat distribution in fuel cells, or molecular structures in X-ray crystallography

8.5.1. CARRS SWEEP ALGORITHMUS

Carr et al. [8]: Sweep Algorithm: Join und Split Tree werden symmetrisch erstellt, nur in gegenläufigen "Sweeping" Richtungen.

(1) Sort all n vertices of the mesh by their function values. (2) Perform a sweep of the n vertices from the smallest function value to the largest function value, and build the join tree. (3) Perform another sweep of the n vertices, now from the largest function value to the smallest function value, and build the split tree. (4) Merge the join tree and split tree together and remove all degree-two nodes in the resulting tree to obtain the contour tree.

Chiang meint, die Zeitbehauptungen von Carr wären leicht unterschiedlich im Paper für (2) und (3). Jedenfalls bestätigt er das Theorem aus [8, Section 4.2]: Ein Konturbaum kann in linearer Zeit vom split und join Baum konstruiert werden.

8.5.2. MONOTONPFADALGORITHMUS

Unterschied zum Sweep: Hier werden nicht alle Vertices n , sondern nur die kritischen Vertices t durchlaufen: Schneller wenn: $t \ll n$.

(1) For constructing the contour tree, identify the component-critical vertices in the mesh. For constructing the contour topology tree, identify all critical vertices in the mesh. (2) Sort those vertices by their function values. (3) Build the join tree: Process those sorted vertices by increasing function values. For each current vertex v_i perform the following: (a) Start a monotone descending path (defined later) from every component in $N(v_i)$ until a vertex which was already visited is found. (b) Connect v_i to the appropriate tree component if v_i is not already connected. (4) Build the split tree: This is symmetric to step (3). Now process the sorted vertices by decreasing function values. (5) Merge the join tree and the split tree together. If the contour topology tree is desired, the algorithm stops here. (6) To get the contour tree, remove the nodes of degree two from the tree.

8.5.3. AUSWERTUNG/VORTEILE GGÜ CARR

Schneller wenn kritische Vertices wesentlich weniger als normale. Braucht weniger Speicherplatz, da nur die kritischen sortiert werden müssen.

8.5.4. BEWEIS FÜR KRITISCHE PUNKTE

Enthält formale Beweisführung für das Theorem über kritische Punkte in einem simplizialem Mesh eines Manifolds (Theorem 1).

8.6. LANEY2006 TURBULENT MIXING LAYER

[Lan+06] Definition Bubble, Zeitverfolgung

[Lan+06], S. 1055 Sohn and Bajaj [2] compute correspondences between contour trees at successive time-steps by using volume matching as in [25] rather than the topological analysis used in [12] and [27]. They use this correspondence to encode the evolution of the isosurface of a fixed isovalue over time in the Topology Change Graph.

9. NOTIZEN SKELETEXTRAKTION UND LASER/ZUSAMMENHANGSKOMPONENTEN

9.1. AU2008SKELETONEXTRACTIONBYMESHCONTRACTION

[Au+08] Extraktion direkt auf dem Mesh ohne Volumenumwandlung: 1) Mesh wird auf volumenlose Skelettform kontrahiert durch implizites Laplacian smoothing (Meshalgorithmus) zusammen mit globalen Positionseinschränkungen. Die Kontraktion erhält die Zusammenhangskomponenten und der Schlüsseleigenschaften des originalen Meshes. 2) Konvertierung in ein 1D Kurvenskelett durch eine Verbindungsoperation. So werden alle zusammengefallen Flächen entfernt und gleichzeitig die Form und Topologie erhalten. 3) Zentrierung des Skeletts wird durch Nutzung des induzierten skeleton-mesh mapping verfeinert.

(2) enthält Informationen über Objektgeometrie, v.a. lokale Dicke.

9.1.1. ANDERE ARBEITEN

- volumetrische Methoden - geometrische Methoden: Voronoi

9.2. KLASING2008EFFICIENTSEGMENTATION

- radialgebundener nächster Partner mit nur zwei Parametern (klingt ähnlich der eigenen Idee für die Zusammenhangskomponente: Tiefe Zentralpartikel und Abstand zu ihm!)

Wenig Quellen über 3D Laserdaten. Hier Segmentation von 3D laser range Daten als preprocessing Schritt über aufwändige Clusteringalgorithmen.

9.2.1. CLUSTERING

- 3D Raum - Cluster sind disjunkt - Distanzmessung zwischen zwei Punkten existiert (euklidisch)
- Graphausdruck: Knoten = Punkte, Kanten = (Punkt 1, Punkt 2, Abstand)

Jeder Knoten ist mit allen Nachbarn verbunden, die in einem vordefiniertem Radius liegen. Liegen weniger als eine minimale Anzahl an Knoten in dem Bereich, wird der Cluster als rauschen definiert.

Graph nur als mathematische Erklärung, in der Programmierung braucht kein Graph erstellt werden: for every point: Aktueller Punkt bereits in einem Cluster? continue zum nächsten Point Alle Nachbarn in einem bestimmten radius finden: Ist einer bereits in einem Cluster -> diesen Punkt zu dem Cluster hinzufügen Sind die anderen Punkte in einem anderen Cluster? Cluster mergen!

Muss nicht für jeden Punkt einen nearest Neighbortest durchführen (Überspringen der bereits zugeordneten Punkte).

9.3. TAGLIASACCHI2012MEANCURVATURESKELETONS

Andere Arbeiten: - Medialaxenskelette - Kurvenskelette - Mittlerer Krümmungsfluss

9.4. WANG2008CURVESKELETONEXTRACTION

[WL08, S. 2] a skeleton is a 1D representation containing zero volume

10. TEMPORÄRE NOTIZEN/OBSOLETES

10.1. ALTE PARAMETERATTRIBUTSUWEISUNG

Aus den Parametern sowie der in [Unterabschnitt 5.1.2](#) genannten Qualität der Unterscheidungsmöglichkeit für die Attribute ergibt sich die in [Tabelle 10.1](#) aufgeführte Empfehlung.

10.2. ALTE MOCKUPS (ILLUSTRATOR)

Weiterhin wird bei Nichtverwendung des Attributs *Größe* ein definierter Durchmesser von 50 [Pixel \(px\)](#) festgelegt, der identisch zur Schrittweite des zugrundeliegenden Rasters ist, um Überlappungen zu vermeiden.

- Durchmesser: [25, 75] [px](#), Standardwert 50[px](#)

Der Durchmesserbereich wurde aufgrund der Darstellungen zugrundeliegenden Rasters mit einer Schrittweite von 50 [px](#) gewählt.

10.3. ZIELSTELLUNG

Molekularsimulationen ermöglichen die Darstellung vieler Partikel über die Zeit. Eine Animation beherbergt allerdings immer das Problem der mangelnden Vergleichbarkeit. Mit OpenGL soll ein Standbild erstellt werden, welche die Clusterbildung, -teilung und das Clustersterben über einen Zeitraum visualisiert.

Tabelle 10.1. Art der Zuweisung zwischen Parametern und Attributen

	Zeitpunkt	Ort	Art
Position	Wert	Wert	Vokabular
Größe	Wert	Wert	Vokabular
Farbe	Wert	Wert	Vokabular
Form	Vokabular	Vokabular	Vokabular
Opazität	Wert	Wert	Vokabular

Tabelle 10.2. Matrix, was wie zusammenpasst und was nicht. Inzwischen überholt!

	Zeitpunkt	Ort	Art (merge, split, death, birth)	örtliche und zeitliche Agglomeration
Position	✓	✓	✓	✓
Größe	✓	✗	✓	✓
Farbe	✓	✗	✓	✓
Form	✗	✗	✓	✓
Opazität	✓	✗	✓	✓

Tabelle 10.3. Zuweisungsmöglichkeiten der Parameter zu den visuellen Attributen bei Kopplung des *Ortes* an die *Position*. Die Häufung H ist nicht Teil.

Attribu- te	Position x	Position y	Position z	Farb- wert & Sätti- gung	Hellig- keit	Ecken- anzahl	Größe	Opazi- tät
Ort x	Ort y	Ort z		Art	Zeit			
Ort x	Ort y	Ort z		Art		Zeit		
Ort x	Ort y	Ort z		Art			Zeit	
Ort x	Ort y	Ort z		Art				Zeit
Ort x	Ort y	Ort z	Zeit		Art			
Ort x	Ort y	Ort z			Art	Zeit		
Ort x	Ort y	Ort z			Art		Zeit	
Ort x	Ort y	Ort z			Art			Zeit
Ort x	Ort y	Ort z	Zeit			Art		
Ort x	Ort y	Ort z			Zeit	Art		
Ort x	Ort y	Ort z				Art	Zeit	
Ort x	Ort y	Ort z						Zeit
Ort x	Ort y	Ort z	Zeit				Art	
Ort x	Ort y	Ort z			Zeit		Art	
Ort x	Ort y	Ort z				Zeit	Art	
Ort x	Ort y	Ort z						Zeit
Ort x	Ort y	Ort z	Zeit					Art
Ort x	Ort y	Ort z			Zeit			Art
Ort x	Ort y	Ort z				Zeit		Art
Ort x	Ort y	Ort z					Zeit	Art

11. AUSBLICK

Weitere Untersuchungsmöglichkeiten:

- Attribut *Position* als *Vokabular* bei örtlicher Häufung von Ereignissen
- Wirkung der Farbzuordnung untersuchen
- Elementattribute (Größe, Opazität) abhängig vom Zoom
- *Form*: Polyeder on-the-fly berechnen für unendlich viele Varianten (bei konvexen: vertex [0f-2f] und edge truncation [0f-1f], siehe Blender/Regular Solids) und offene Formen erlauben (keine Flächen, nur dicke Kanten), vgl. [Gra15].
- Code: Call Eventeinteilung nach Typ, evtl auch Zeit (weniger Speicherplatz).
- Nutzung von Animationen mit visuellen Bewegungsattributen (Bewegungsrichtung, Bewegungsbahn, Beschleunigung, Geschwindigkeit) sowie Veränderung der anderen Attribute; die Zuweisung der Bewegungsattribute wäre an die Parameter Zeit und Position denkbar, die Zuweisung der Veränderung der anderen Attribute kann an alle drei Parameter erfolgen

Benutzung von flexiblen Isooberflächen nach [CSP10].

A. ANHANG

A.1. BERECHNUNGEN MOCKUPS

WERTE VERALTET BZW. ZEIGEN DER TRIVIALEN BERECHNUNG SINNLOS/KANN GELÖSCHT WERDEN

In [Gleichung A.1](#) ist die Berechnung des Farbwerts h in linearer Abhängigkeit von der Zeit t zu finden.

$$\begin{aligned} h[^\circ] &= m \cdot t + n \\ h_{\max} &= m \cdot t_{\max} + n && \text{mit } h_{\max} = 320, t_{\max} = 23 \\ h_{\min} &= m \cdot t_{\min} + n && \text{mit } h_{\min} = 0, t_{\min} = 1 \\ n &= -14, \overline{54} \\ m &= 14, \overline{54} \\ h[^\circ] &= \boxed{[14, \overline{54} \cdot t + 14, \overline{54} + 0, 5]} \end{aligned} \tag{A.1}$$

In [Gleichung A.2](#) wird die Berechnung der Helligkeit b in linearer Abhängigkeit von der Zeit t gezeigt.

$$\begin{aligned} b[\%] &= m \cdot t + n \\ b_{\max} &= m \cdot t_{\max} + n && \text{mit } b_{\max} = 100, t_{\max} = 23 \\ b_{\min} &= m \cdot t_{\min} + n && \text{mit } b_{\min} = 40, t_{\min} = 1 \\ n &= 37, \overline{27} \\ m &= 2, \overline{72} \\ b[\%] &= \boxed{[2, \overline{72} \cdot t + 37, \overline{27} + 0, 5]} \end{aligned} \tag{A.2}$$

In [Gleichung A.3](#) ist die Berechnung der Durchmesser d in linearer Abhängigkeit von der Zeit t zu finden.

$$\begin{aligned} d[px] &= m \cdot t + n \\ d_{\max} &= m \cdot t_{\max} + n && \text{mit } d_{\max} = 75, t_{\max} = 23 \\ d_{\min} &= m \cdot t_{\min} + n && \text{mit } d_{\min} = 25, t_{\min} = 1 \\ n &= 22, \overline{72} \\ m &= 2, \overline{27} \\ d[px] &= \boxed{[2, \overline{27} \cdot t + 22, \overline{72} + 0, 5]} \end{aligned} \tag{A.3}$$

Tabelle A.1. Die Beispieldatensätze für die Mockups. Die x-Koordinate des Ortes ist mit dem Zeitpunkt t gekoppelt über $x = \lfloor \frac{4}{3} \cdot t + 0,5 \rfloor$. Die y, z-Koordinaten sind, bis auf wenige Ausnahmen zur Demonstration der Häufung, zufällig verteilt im Intervall [1, 10].

Datensatznummer	Zeitpunkt t	Ort (x, y, z)	Art
1	1	1, 8, 4	Split
2	1	1, 4, 3	Split
3	1	1, 1, 8	Split
4	1	1, 4, 1	Split
5	1	1, 4, 9	Split
6	1	1, 4, 4	Split
7	1	1, 4, 4	Merge
8	1	1, 4, 4	Birth
9	1	1, 4, 4	Death
10	1	1, 4, 6	Split
11	2	3, 4, 1	Split
12	2	3, 8, 7	Birth
13	2	3, 7, 8	Split
14	2	3, 6, 1	Split
15	2	3, 1, 2	Merge
16	2	3, 5, 8	Split
17	3	4, 3, 1	Merge
18	3	4, 10, 10	Merge
19	3	4, 5, 4	Merge
20	3	4, 3, 1	Merge
21	4	5, 5, 7	Merge
22	4	5, 3, 1	Split
23	4	5, 3, 10	Merge
24	4	5, 1, 1	Split
25	5	7, 9, 1	Merge
26	5	7, 4, 2	Merge
27	6	8, 3, 9	Split
28	6	8, 10, 10	Death
29	6	8, 1, 5	Split
30	6	8, 8, 1	Death
31	7	9, 3, 5	Death
32	7	9, 6, 4	Death
33	8	11, 3, 8	Split
34	8	11, 3, 2	Split
35	10	13, 2, 7	Merge
36	10	13, 2, 7	Split
37	10	13, 10, 5	Merge
38	10	13, 2, 2	Split
39	13	17, 8, 1	Merge
40	13	17, 1, 8	Merge
41	15	20, 5, 3	Split
42	15	20, 7, 6	Split
43	16	21, 7, 3	Death
44	16	21, 6, 6	Merge
45	18	24, 8, 4	Merge
46	20	27, 6, 5	Merge
47	20	27, 6, 5	Merge
48	20 ³⁷	27, 6, 5	Death
49	21	28, 5, 4	Merge
50	21	28, 5, 3	Merge

Tabelle A.2. Festlegung der Attributeigenschaften für den Parameter *Zeit* in den Mockups.
VERALTET

Zeit	Farbwert [°]	Helligkeit [%]	Anzahl Ecken	Größe [px]
1	0	40	1	25
2	15	43	2	27
3	29	45	3	30
4	44	48	4	32
5	58	51	5	34
6	73	54	6	36
7	87	56	7	39
8	102	59	8	41
10	131	65	10	45
13	175	73	13	52
15	204	78	15	57
16	218	81	16	59
20	276	92	20	68
21	291	95	21	70
23	320	100	23	75

Analog erfolgen die Berechnungen für den Parameter *Ort x* mit den Grenzwerten

$$\begin{aligned}x_{\min} &= 1 \\x_{\max} &= 31\end{aligned}\tag{A.4}$$

Das Ergebnis ist in [Tabelle A.3](#) zu finden.

A.2. WERKZEUGE UND BIBLIOTHEKEN

MOCKUP VISUALISIERUNG

Adobe Illustrator
Blender mit Regular Solids
Unity3D mit [UIComboBox](#), [Simple FlyCamera](#)

DOKUMENTATION

Adobe Photoshop
Blender mit Extra Objects
[TeXstudio](#), [TeXlive](#) mit tudscr, glossaries, tikz u.a.

Tabelle A.3. Festlegung der Attributeigenschaften für den Parameter *Ort x* in den Mockups.
VERALTET

Ort x	Farbwert [°]	Helligkeit [%]	Anzahl Ecken	Größe [px]
1	0	40	1	25
3	21	44	3	28
4	32	46	4	30
5	43	48	5	32
7	64	52	7	35
8	75	54	8	37
9	85	56	9	38
11	107	60	11	42
13	128	64	13	45
17	171	72	17	52
20	203	78	20	57
21	213	80	21	58
27	277	92	27	68
28	288	94	28	70
31	320	100	31	75

LITERATUR

- [All+15] Andy Allan u. a. *Contours*. Englisch. 2015. URL: <http://wiki.openstreetmap.org/wiki/Contours> (besucht am 23.05.2015).
- [Au+08] Oscar Kin-Chung Au u. a. *Skeleton Extraction by Mesh Contraction*. Englisch. 2008. URL: <http://www.math.tau.ac.il/~dcor/articles/2008/Skeleton-Extraction.pdf> (besucht am 10.02.2015).
- [Bai04] Rudolf H. Baierl. *Konvexe Polyeder: Klassische und hybride numerische Generierungsmethoden*. Beuth Hochschule für Technik Berlin. 2004. URL: http://public.beuth-hochschule.de/~baierl/inhalt/polyhed/p_konst000530.pdf (besucht am 06.03.2015).
- [Blu67] Harry Blum. „A Transformation for Extracting New Descriptors of Shape“. In: *Models for the Perception of Speech and Visual Form*. Hrsg. von Weiant Wathen-Dunn. Cambridge: MIT Press, 1967, S. 362–380.
- [Bog15] Alexander Bogomolny. *Regular Polyhedra*. Englisch. Interactive Mathematics Miscellany und Puzzles. 2015. URL: http://www.cut-the-knot.org/do_you_know/polyhedra.shtml (besucht am 06.03.2015).
- [Chi+05] Yi-Jen Chiang u. a. „Simple and optimal output-sensitive construction of contour trees using monotone paths“. In: *Computational Geometry* 30.2 (2005). Special Issue on the 19th European Workshop on Computational Geometry, S. 165–195. ISSN: 0925-7721. DOI: <http://dx.doi.org/10.1016/j.comgeo.2004.05.002>. URL: <http://www.sciencedirect.com/science/article/pii/S0925772104000811>.
- [CLM54] H. S. M. Coxeter, M. S. Longuet-Higgins und J. C. P. Miller. „Uniform Polyhedra“. In: *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 246.916 (1954), S. 401–450. ISSN: 0080-4614. DOI: [10.1098/rsta.1954.0003](https://doi.org/10.1098/rsta.1954.0003).
- [Cor+05] Nicu D Cornea u. a. „Computing hierarchical curve-skeletons of 3D objects“. In: *The Visual Computer* 21.11 (2005), S. 945–955.
- [CS09] Hamish Carr und Jack Snoeyink. „Representing Interpolant Topology for Contour Tree Computation“. English. In: *Topology-Based Methods in Visualization II*. Hrsg. von Hans-Christian Hege, Konrad Polthier und Gerik Scheuermann. Mathematics and Visualization. Springer Berlin Heidelberg, 2009, S. 59–73. ISBN: 978-3-540-88605-1. DOI: [10.1007/978-3-540-88606-8_5](https://doi.org/10.1007/978-3-540-88606-8_5). URL: http://dx.doi.org/10.1007/978-3-540-88606-8_5.

- [CSA01a] Hamish Carr, Jack Snoeyink und Ulrike Axen. *Computing Contour Trees in All Dimensions*. Englisch. 2001. URL: <http://www.csi.ucd.ie/staff/hcarr/home/research/publications/contourtreealgorithm.pdf> (besucht am 16.03.2015).
- [CSA01b] Hamish Carr, Jack Snoeyink und Ulrike Axen. *Efficient computation of Contour Trees*. Englisch. 2001. URL: <http://www.csi.ucd.ie/staff/hcarr/home/research/contourtrees/contourtrees.html> (besucht am 16.03.2015).
- [CSM07] Nicu D Cornea, Deborah Silver und Patrick Min. „Curve-skeleton properties, applications, and algorithms“. In: *Visualization and Computer Graphics, IEEE Transactions on* 13.3 (2007), S. 530–548.
- [CSP10] Hamish Carr, Jack Snoeyink und Michiel van de Panne. „Flexible isosurfaces: Simplifying and displaying scalar topology using the contour tree“. In: *Computational Geometry* 43.1 (2010). Special Issue on the 14th Annual Fall Workshop, S. 42–58. ISSN: 0925-7721. DOI: <http://dx.doi.org/10.1016/j.comgeo.2006.05.009>. URL: <http://www.sciencedirect.com/science/article/pii/S0925772109000455>.
- [DS06] Tamal K. Dey und Jian Sun. „Defining and Computing Curve-skeletons with Medial Geodesic Function“. In: *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*. SGP '06. Cagliari, Sardinia, Italy: Eurographics Association, 2006, S. 143–152. ISBN: 3-905673-36-3. URL: <http://dl.acm.org/citation.cfm?id=1281957.1281975>.
- [Gra15] Sam Gratrix. *WebGL Uniform Polyhedra*. Englisch. 2015. URL: <http://gratrix.net/polyhedra/webgl/> (besucht am 06.03.2015).
- [Gro+07] Sebastian Grottel u. a. „Visual Verification and Analysis of Cluster Detection for Molecular Dynamics“. In: *Proceedings of IEEE Visualization '07*. Bd. 13. 6. 2007, S. 1624–1631. DOI: [10.1109/TVC.2007.70614](https://doi.acm.org/10.1109/TVC.2007.70614). URL: <http://doi.acm.org/10.1109/TVC.2007.70614>.
- [Gro+09] Sebastian Grottel u. a. „Topological Extraction and Tracking of Defects in Crystal Structures“. Englisch. In: Springer, 2009, S. 167–178. URL: http://www.visus.uni-stuttgart.de/uploads/tv_vispublications/topoinvis09-grottel.pdf (besucht am 10.02.2015).
- [Gro14] Sebastian Grottel. *File Format Specification MMPLD*. Englisch. 2014. URL: <https://cloud.visus.uni-stuttgart.de/public.php?service=files&t=b859555761a09c5857a705b&download=> (besucht am 28.04.2015).
- [Har08] George W. Hart. *Procedural Generation of Sculptural Forms*. Englisch. Computer Science Department, Stony Brook University. 2008. URL: <http://www.georgehart.com/ProceduralGeneration/Bridges08-Hart10pages.pdf> (besucht am 06.03.2015).
- [HF05] M Sabry Hassouna und Aly A Farag. „Robust centerline extraction framework using level sets“. In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Bd. 1. IEEE. 2005, S. 458–465.
- [Hop96] Edward J. Hopkins. *Surface Weather Analysis Chart*. Englisch. 1996. URL: <http://www.meteor.wisc.edu/~hopkins/aos100/sfc-anl.htm> (besucht am 20.05.2015).
- [Hur10] Lorenz Hurni. „Cartographic Relief Presentation Revisited - Forty Years after Eduard Imhof“. English. In: *Landform - Structure, Evolution, Process Control*. Hrsg. von Jan-Christoph Otto und Richard Dikau. Bd. 115. Lecture Notes in Earth Sciences. Springer Berlin Heidelberg, 2010, S. 1–20. ISBN: 978-3-540-75760-3. DOI: [10.1007/978-3-540-75761-0_1](https://doi.org/10.1007/978-3-540-75761-0_1). URL: http://dx.doi.org/10.1007/978-3-540-75761-0_1.
- [Joh66] Norman W. Johnson. „Convex Solids with Regular Faces“. In: *Canadian Journal of Mathematics* 18 (1966), S. 169–200. ISSN: 0008-414x.

- [KT03] Sagi Katz und Ayellet Tal. *Hierarchical mesh decomposition using fuzzy clustering and cuts*. Bd. 22. 3. ACM, 2003.
- [Lan+06] D. Laney u. a. „Understanding the Structure of the Turbulent Mixing Layer in Hydrodynamic Instabilities“. In: *Visualization and Computer Graphics, IEEE Transactions on* 12.5 (Sep. 2006), S. 1053–1060. ISSN: 1077-2626. DOI: [10.1109/TVCG.2006.186](https://doi.org/10.1109/TVCG.2006.186).
- [MWL02] Cherng-Min Ma, Shu-Yen Wan und Jiann-Der Lee. „Three-dimensional topology preserving reduction on the 4-subfields“. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24.12 (Dez. 2002), S. 1594–1605. ISSN: 0162-8828. DOI: [10.1109/TPAMI.2002.1114851](https://doi.org/10.1109/TPAMI.2002.1114851).
- [Pal08] Kálmán Palágyi. „A 3D fully parallel surface-thinning algorithm“. In: *Theoretical Computer Science* 406.1–2 (2008). Discrete Tomography and Digital Geometry: In memory of Attila Kuba, S. 119–135. ISSN: 0304-3975. DOI: <http://dx.doi.org/10.1016/j.tcs.2008.06.041>. URL: <http://www.sciencedirect.com/science/article/pii/S0304397508004350>.
- [Pas+07] Valerio Pascucci u. a. „Robust on-line computation of Reeb graphs: simplicity and speed“. In: *ACM Transactions on Graphics (TOG)*. Bd. 26. 3. ACM. 2007, S. 58.
- [Tag+12] Andrea Tagliasacchi u. a. „Mean curvature skeletons“. In: *Computer Graphics Forum*. Bd. 31. 5. Wiley Online Library. 2012, S. 1735–1744.
- [Tan+14] San Tang u. a. „Segmentation and 3D visualization of pheochromocytoma in contrast-enhanced CT images“. In: *Audio, Language and Image Processing (ICALIP), 2014 International Conference on*. Juli 2014, S. 39–43. DOI: [10.1109/ICALIP.2014.7009753](https://doi.org/10.1109/ICALIP.2014.7009753).
- [Web+07] Ofir Weber u. a. „Context-Aware Skeletal Shape Deformation“. In: *Computer Graphics Forum*. Bd. 26. 3. Wiley Online Library. 2007, S. 265–274.
- [Wei15] Eric W. Weisstein. *Kepler-Poinsot Solid*. Englisch. MathWorld—A Wolfram Web Resource. 2015. URL: <http://mathworld.wolfram.com/Kepler-PoinsotSolid.html> (besucht am 06. 03. 2015).
- [WL08] Yu-Shuen Wang und Tong-Yee Lee. „Curve-Skeleton Extraction Using Iterative Least Squares Optimization“. In: *Visualization and Computer Graphics, IEEE Transactions on* 14.4 (Juli 2008), S. 926–936. ISSN: 1077-2626. DOI: [10.1109/TVCG.2008.38](https://doi.org/10.1109/TVCG.2008.38).
- [WPP07] Robert Y Wang, Kari Pulli und Jovan Popović. „Real-time enveloping with rotational regression“. In: *ACM Transactions on Graphics (TOG)* 26.3 (2007), S. 73.

INDEX

Agglomeration, 17, 21
archimedischer Körper, 10
Art, 20, 21

Blender, 21

catalanischer Körper, 10

Farbe, 16, 21

Form, 16, 20, 21, 35

Gesetz der Ähnlichkeit, 16

Gesetz der fortgesetzt durchgehenden Linie, 17

Gesetz der gemeinsamen Bewegung, 16

Gesetz der gemeinsamen Region, 16

Gesetz der Geschlossenheit, 16, 17

Gesetz der Gleichzeitigkeit, 16

Gesetz der Kontinuität, 16

Gesetz der Nähe, 16

Gesetz der Prägnanz, 16

Gesetz der verbundenen Elemente, 17, 19

Größe, 16, 20, 33

Häufung, 17–19, 21, 23, 34

Isolinie, 8

Isooberfläche, 8

Isowert, 8

Johnson-Körper, 10

Kepler-Poinsot-Körper, 10

Kontur, 8, 13

MegalMol, 18

MegaMol, 17, 18, 20

Opazität, 18, 19, 21, 23

Ort, 18–20, 34, 38, 39

platonischer Körper, 10, 21

Position, 15–19, 34, 35

Saatpartikel, 13

Tiefe, 13

Tiefenwert, 8, 13, 14

Unity3D, 19, 21

Vokabular, 15, 35

Zahlenwert, 15

Zeit, 19, 38