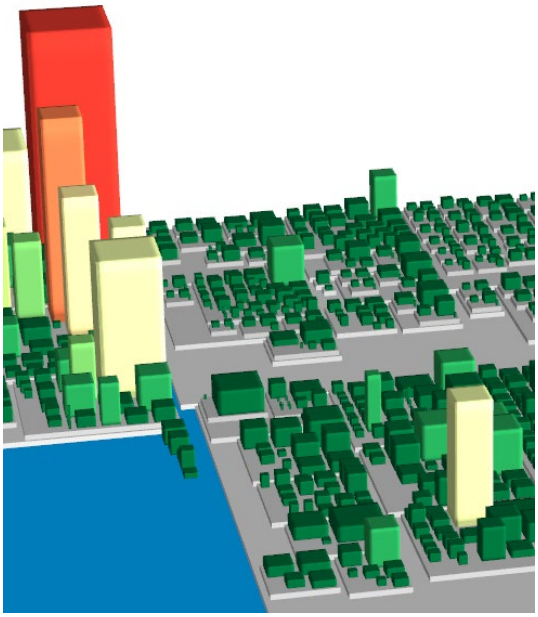
1 – Tree Maps



Beschreibung Tree Maps bilden hierarchisch geordnete Daten mit Hilfe einer Baumstruktur ab. Jeder Zweig wird als Rechteck dargestellt, welche selbst in kleinere Rechtecke unterteilt wird, um da- runterliegende Zweige anzuzeigen. Ein Blattknoten, dargestellt durch ein Rechteck, hat selbst eine Fläche proportional zu einer gewünschten Dimension eines Datums wie zum Beispiel der Dateigröße. Mittels Farb-codierung lassen sich zusätzlich zum Beispiel Dateitypen unterscheiden.

Software, Literatur

- JavaView
- Ben Shneiderman: „Tree visualization with Tree-maps: A 2d space-filling approach" (FU Bibliothek)

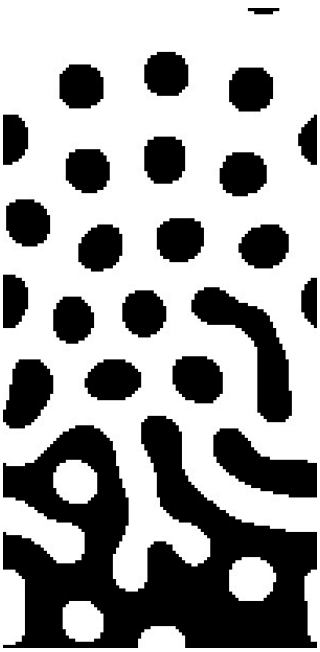
Erwartung

- Implementieren Sie 3D Tree Maps in JavaView, wobei Sie durch die Höhe eine geeignete Information visualisieren.
- Bearbeiten Sie zwei der vier folgenden Fragestellungen.

Fragestellungen

- Welche Vorteile/Nachteile sehen Sie in den Tree Maps im Vergleich zu anderen Visualisierungen von z.B. Festplatteninhalten? Welche Vorteile/Nachteile ergeben sich spezifisch beim Übergang von 2D zu 3D?
- Welche Ideen haben Sie, die Tree Maps in Bezug auf ihre Nachteile (s. obige Frage) zu verbessern? Implementieren Sie Ihre Ideen.
- Wie kann man mit Hilfe von Tree Maps die folgenden Abfragen beantworten und wie ist dies möglichst effizient möglich? Welche Zeit wird für die Abfragen benötigt?
    - Welches Datum belegt den größten Platz in der Hierarchie?
    - Welcher Ordner belegt den größten Platz in der Hierarchie?
    - Welcher Ordner/welches Datum auf Ebene X belegt den größten Platz in der Hierarchie?
- Überlegen Sie, welche Zeit es benötigt, um eine Tree Map in Abhängigkeit der Menge der Dateien zu bauen. (Landau-/O-Notation)

2 – Turing-ähnliche Muster basierend auf zellulären Automaten



Beschreibung Turing, Young und McCabe untersuchten die Formation von Hautmustern bei Tieren. Nach Young (1984) kann ein solches Muster durch den zellulären Automaten modelliert werden. Einige Parameter des Automaten können dabei variiert werden wie zum Beispiel der Stempel bzw. die Nachbarschaftsform, auf dem/der sie agieren.

Software, Literatur

- Gary Greenfield: „Turing-like Patterns from Cellular Automata"[1]
- Biological Patterns: Turing and Young[2]
- Allen B. Downey: „Think Complexity" (FU Bibliothek)

Erwartungen

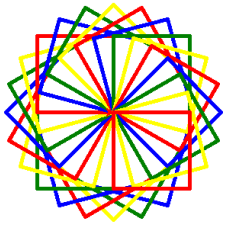
- Implementieren Sie den zellulären Automaten.
- Bearbeiten Sie die folgenden Fragestellungen.

Fragestellungen

- Der zelluläre Automat modelliert ein diskretes dynamisches System, welches die Entwicklung von Zellen zum neuen Zeitpunkt bezüglich der Nachbarschaft des alten Zeitpunkts gemäß bestimmter Regeln abbildet. Überlegen Sie sich:
    - eigene Regeln (Überführungsfunktion),
    - eine eigene (ggf. nicht symmetrische) Nachbarschaft
    - und eine eigene Zustandsmenge (z.B. tot/komatös/lebendig).
- Normalerweise wird der zelluläre Automat auf der Fläche eines Torus ausgeführt, d.h. der rechte und linke sowie der obere und untere Rand sind identifiziert. Was passiert, wenn man den Automaten auf einem Zylinder (nur linker und rechter Rand identifiziert) oder einem Rechteck ausführt? Wie müssen die Regeln für Zellen am Rand geändert werden? Welche Randbedingungen können angelegt werden?

[1] http://archive.bridgesmathart.org/2016/bridges2016-151.pdf
[2] https://the-biologist-is-in.blogspot.de/2015/12/biological-patterns-turing-and-young.html

3 – Turtle Graphics



Beschreibung Turtle Graphics sind Vektorgraphiken, die einen relativen Cursor („Turtle“) in der Ebene verwenden. Die „Turtle“ hat drei Kenngrößen: ‚Ort‘, ‚Ausrichtung‘ und ‚Stift‘. Der Stift wiederum verfügt über die Kenngrößen ‚Farbe‘, ‚Breite‘ und ‚Zustand‘ (an/aus). Die „Turtle“ bewegt sich relativ zu ihrer Position anhand von Befehlen wie „10 Einheiten in Richtung (x,y)“ oder „Drehen um 90° nach links“. Neben spirographenartigen Kurven kann man die „Turtle“ *Lindenmayer-Systeme* visualisieren lassen.

Software, Literatur

- P. Prusinkiewicz, „The Algorithmic Beauty of Plants“, Kapitel 1 online erhältlich unter: http://algorithmicbotany.org/papers/abop/abop-ch1.pdf
- L. Kari et al., „L Systems“, Springer, 2013, S. 253-288

Erwartungen

- Programmieren Sie eine graphische Oberfläche, in der der Nutzer die Kenngrößen der „Turtle“ als Folge eingeben kann.
- Ermöglichen Sie es, die „Turtle“ ein vom Nutzer vorgegebenes Lindenmayer-System visualisieren zu lassen.
- Bearbeiten Sie eine der folgenden Fragestellungen:

Fragestellungen

- Erweitern Sie die graphische Oberfläche derart, dass auch die Stiftparameter eingegeben werden können.
- Ändern Sie den Bewegungsraum der „Turtle“ zu einem Zylinder.[3]

[3] Hier müssen nur die Bewegungen der „Turtle“ implementiert werden, aber nicht die Lindenmayer-Systeme.

4 – Städtegenerierung

Beschreibung Die Erstellung virtueller Welten wird mit steigender Komplexität und höheren Ansprüchen immer zeit- und kostenintensiver. Dies betrifft unter anderem die Videospielindustrie. Eine Lösung bietet das Verfahren der Städtegenerierung, welche unter bestimmten Prämissen zufällig erzeugte Umgebungen generiert.

Software, Literatur

- G. Kelly, H. McCabe: „A Survey of Procedural Techniques for City Generation"
- Stichwort: Algorithmen zur Labyrintherzeugung oder Agent-basierter Ansatz

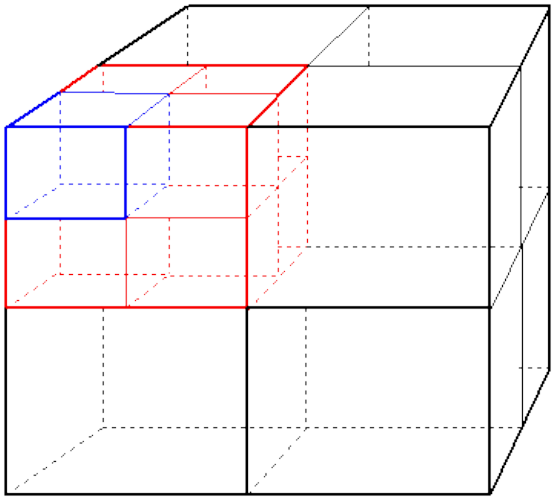
Erwartungen

- Gegeben seien drei verschiedene Kachel-Typen (Straße, Haus, Natur). Implementieren Sie einen 2-dimensionalen Städtegenerator, der als Parameter die Größe des Rasters aufnimmt. Die Generierung soll zufällig erfolgen, jedoch unter der Bedingung, dass ein Straßennetz vorhanden ist, sowie fünf weitere, von Ihnen als sinnvoll erachtete Regeln (z.B. Straßen enden nicht im Nichts oder jedes Haus steht an mindestens einer Straße) befolgt werden.
- Bearbeiten Sie zwei der drei folgenden Fragestellungen.

Fragestellungen

- Implementieren Sie eine weitere Kachelform (z.B. Schiene oder Wasser).
- Passen Sie Ihren Algorithmus auf manuelle Eingaben hin an (z.B. die Vorgabe von Regionen wie Wasser oder Gebäude).
- Überlegen Sie sich Randbedingungen für Ihre Generierung oder implementieren Sie diese auf dem Torus oder der 2-Sphäre.

5 – Visualisierung von Octrees



Beschreibung Soll ein Computer Datenmengen – durch Punkte repräsentiert – durchsuchen können, ist es notwendig, diese Daten sortiert vorzuhalten. Denn nur so kann sichergestellt werden, dass auf der Suche nach einem bestimmten Datensatz nicht alle Punkte angeschaut werden müssen und die entsprechende Abfragen schnell ablaufen. Eine Datenstruktur, die hier helfen kann, sind Octrees.

Software, Literatur

- M. de Berg et al. „Computational Geometry“, Springer, 2007.

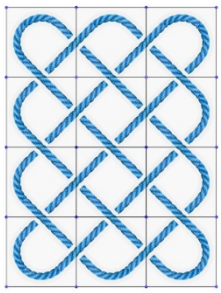
Erwartungen

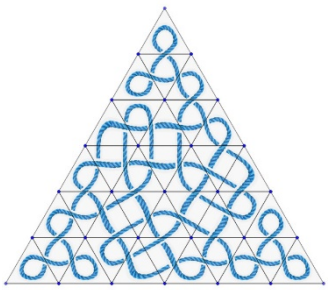
- Implementieren Sie die Datenstruktur von Octrees für 2- und 3- dimensionale Punktmengen.
- Visualisieren Sie die Datenstruktur für 2-dimensionale Punktmengen.
- Bearbeiten Sie drei der fünf folgenden Fragestellungen.

Fragestellungen

- Implementieren Sie eine Visualisierung für 3-dimensionale Punktmengen.
- Geben Sie dem/der Nutzer*in die Möglichkeit, den Octree nur bis zu einer definierten Tiefe anzuzeigen.
- Machen Sie die Anzahl von Punkten, die in einem Blatt des Octrees gespeichert werden, variabel.
- Implementieren Sie eine Visualisierung, welche nur diejenigen Boxen anzeigt, die für einen von/vom der/dem Nutzer*in Punkt der Punktmenge relevant sind.
- Trennen Sie die Punktmenge nicht in der Mitte auf, sondern implementieren Sie eine neue Auftrennung, welche den Octree für eine gegebene Punktmenge möglichst balanciert erstellt.

6 – Texturabbildungen





Beschreibung In der Visualisierung werden Texturabbildungen benutzt, um Flächen (im $\mathbb{R}^3$) mit 2D-Graphiken (Texturen) wie mit einer Tapete zu überdecken. In diesem Projekt sollen Texturabbildungen verwendet werden, um auf einem regelmäßigen Gitter in der Ebene Knoten bzw. Gewebe zu visualisieren. Diese können durch bestimmte Operationen auf den Kanten abwechslungsreicher gestaltet werden.

Software, Literatur

- JavaView
- "Théorie des nœuds et enluminure celte", Christian Mercat[4]
- "A Celtic Framework for Knots and Links", Jonathan L. Gross and Thomas W. Tucker[5]
- "Computer Generated Celtic Design", Matthew Kaplan and Elaine Cohen[6]

Erwartungen

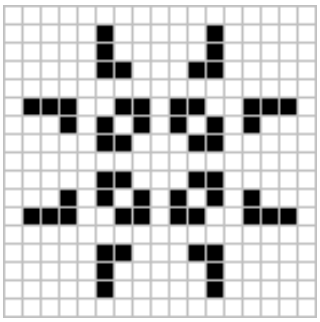
- Erzeugen Sie auf einem regelmäßigen Dreiecks- und Viereckgitter in der Ebene einen regulären Knoten wie in "A Celtic Framework for Knots and Links" beschrieben.
- Stellen Sie den Knoten durch eine passende Texturabbildung auf den Dreiecken bzw. Vierecken dar (z.B. mit JavaView, openGL, webGL, ...).
- Bearbeiten Sie die folgenden Aufgabenstellungen:

Fragestellungen

- Färben Sie die einzelnen Zusammenhangskomponenten des Knotens unterschiedlich ein.
- Implementieren Sie die Kantenoperationen aus "Théorie des nœuds et enluminure celte".

[4] https://mathinfo.unistra.fr/websites/math-info/irem/Publications/L_Ouvert/n084/o_84_1-22.pdf
[5] www.cs.columbia.edu/~gross/research/CeltFrameGT-DCG.pdf
[6] www.matt-kaplan.com/npr/knots/Knot_elec.pdf

7 – Game of Life



Beschreibung Das „Game of Life“ von John Conway ist ein zellulärer Automat, der mit sehr einfachen Regeln „Leben“ simulieren soll; auf einem regulären Gitter werden auf eine Anfangskonfiguration iterativ auf alle Zellen die Regeln angewendet und der Lebenszustand aller Zellen neu bestimmt.
Die Standardregeln sind so abgestimmt, dass das „Überleben“ der Population schwer vorhersagbar ist.

Software, Literatur

- Martin Gardner: Mathematical Games – The fantastic combinations of John Conway's new solitaire game "life"[7]

Erwartungen

- Implementieren Sie den zellulären Automaten mit zufälligen und editierbaren Anfangskonstellationen.
- Bearbeiten Sie zwei der drei folgenden Fragestellungen.

Fragestellungen

- Wie verhält es sich bei der Schach-Springer-Nachbarschaft?
- Welche Regeln funktionieren auf einem regulären Dreiecks- oder Sechseckgitter? Was passiert bei einem torischen Gitter?
- Experimentieren Sie mit verschiedenen „Lebensformen“ (z.B. blaue und rote „lebendige“ Zellen), Mehrspielermodus, …

[7] https://web.stanford.edu/class/sts145/Library/life.pdf