Exposé zur Bachelorarbeit von Adrian Helberg bei Prof. Dr. Jenke

29. September 2020

1 Problemstellung

Effizientes Objektdesign und -modellierung sind entscheidende Kernkompetenzen in verschiedenen Bereichen der digitalen Welt. Da die Erstellung geometrischer Objekte unintuitiv ist und ein großes Maß an Erfahrung und Expertise erfordert, ist dieses stetig wachsende Feld für Neueinsteiger nur sehr schwer zu erschließen. Die Forschung liefert hierzu einige Arbeiten zu prozeduraler Modellierung, um digitale Inhalte schneller und automatisiert zu erstellen. Die Bachelorarbeit soll sich auf die Erkenntnisse einer Basisquelle [4] stützen und im dreidimensionalen Kontext die inverse prozedurale Modellierung von Verzweigungsstrukturen beschreiben. Das in der Basisquelle beschriebene System generiert zweidimensionale Objekte mit Verzweigungsstrukturen. Hierzu soll ein prototypische Implementierung zur Überführung des Algorithmus vom zweidimensionalen in den dreisimensionalen Raum implementiert werden.

1.1 Konzepte und Ideen

Im Folgenden wird auf die Methodik zum Ableiten eines L-Systems, das eine zweidimensionale Verzweigungsstrukturen repräsentiert der Basisquelle [4] eingegangen und Konzepte und eigene Lösungsansätze präsentiert.

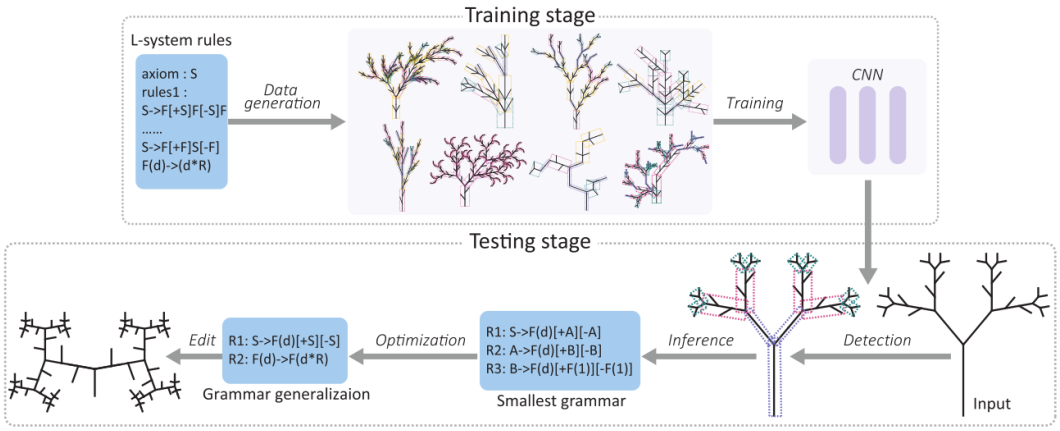

Abbildung 1: Architektur des Systems der Basisquelle [4]

1.1.1 System

Ziel des Systems ist die Erstellung von Ähnlichkeitsabbildern eines Input-Bildes. Hierzu wird eine beliebig verschachtelte Verzweigungsstruktur eingelesen, um eine Sammlung an ähnlichen Strukturen auszugeben.

1.1.2 Voraussetzungen

Das Input-Bild kann als Sammlung verschiedener, atomarer Stukturen (=**Templates**) gesehen werden, die mit diversen Parametern transformiert wurden. Um diese Strukturen zu erkennen, wird ein neuronales Netz anhand von generierten Trainingsdaten angelernt. Die Trainingsdaten werden anhand vordefinierter Templates, eines vordefinierten L-Systems und randomisierten Parametern generiert.
Die in Input-Bildern erwarteten Verzweigungsstrukturen bilden die Templates, die jeweils an einen Schlüssel (=**Label**) gebunden sind.
Um eine Zeichenkette, die Symbole eines L-Systems beinhaltet, zu visualisieren wird ein **Logo-Turtle-Algorithmus** genutzt [5].

1.1.3 Überblick

Abbildung 1 zeigt, wie das o.g. System aufgebaut ist:

I. Training: Ein neuronales Netz (**CNN**), das zur Erkennung von Verzweigungsstrukturen benötigt wird, wird trainiert.

   a) **Datengenerierung**: Mit dem vordefinierten L-System, einer zufälligen Auswahl an Templates und zufälligen Werten für Parameter werden Testdaten in Form von Bilden, die mit dem Logo-Turtle-Algorithmus gezeichnet werden, generiert.

   b) **Training**: Die generierten Bilder werden als Input in ein neuronales Netz gespeist

II. Testing: Die Grammatik über Ähnlichkeitsabbilder des Input-Bildes wird abgeleitet.

   a) **Erkennung**: Das trainierte CNN ist in der Lage Strukturen eines unbekannten Input-Bildes zu erkennen und einem Schlüssel zuzuordnen

   b) **Inferenz**: Die erkannten Strukturen werden in einer Baumtopologie organisiert mit einem Startsymbol als Wurzelknoten, Terminalen als Knoten und Nicht-Terminalen als Blattknoten. Die Kanten des Baumes bilden die jeweils relative Transformation zum Elternknoten. Aus der Baumstruktur kann eine möglichst „kleine“ Grammatik, die nur das Input-Bild beschreibt, abgeleitet werden.

   c) **Optimierung**: Die kompakte Grammatik wird um Rekursionen und nicht-deterministische Regeln erweitert, um ein L-System zu konstruieren, das nicht nur dem Input-Objekt entspricht, sondern eine größere Vielfalt repräsentiert. Anschließend werden ähnliche Regeln zu neuen Symbolen der Grammatik verbunden (**Generalisiserung**).

   d) **Bearbeitung**: Die Parameter der generalisierten Grammatik können vom Benutzer verändert werden, um das Ergebnis des Systems zu beeinflussen.

1.1.4 Umsetzung

Um das beschriebene System vom zweidimensionalen in den dreidimensionalen Raum zu überführen, sind einige Schritte nötig:

- Das vordefinierte L-System muss Objekte im dreidimensionalen Raum erstellen
- Der Logo-Turtle-Algorithmus erhält eine weitere Dimension, um Objekte im 3D darstellen zu können. Derzeit gibt es eine 3D-Turtle-Grafik-Bibliothek [2] für Java
- Die Struktur des neuronalen Netzes verändert sich, da sich die Trainingsdaten ändern.
- Evtl. sind „Schnappschuss"- Bilder des generierten 3D-Objektes aus verschiedenen Kameraperspektiven nötig, damit das CNN Verzweigungsstrukturen im dreidimensionalen Raum erkennen kann [6].
- Begrenzungskörper (**bounding boxes**), die als Schlüssel verwendet werden, müssen dreidimensional sein.
- Evtl. müssen Metriken zu Abstandsberechnung zweier Strukturen und heuristische Verfahren, wie der Greedy-Algorithmus zur Minimierung der Kostenfunktion bei der Generalisierung eines L-Systems, dementsprechend angepasst werden

1.2 Einführung

Die Einführung des Themas könnte wie folgt aussehen:

Mit der Digitalisierung der Welt erlangt die Erstellung von digitalen Inhalten, wie 3D Modelle für Computerspiele, Webdesigns oder Visualisierung von Architektur, zunehmend an Bedeutung. Darum werden Verfahren gesucht, um Objekte dieser Felder formal zu beschreiben und somit kodifizierbar (*engl. codify*) zu machen. Hierbei bilden sich folgende Klassen heraus:

- **Modellierung**: Um einen physikalischen Körper in ein digitales Objekt zu überführen, wird mithilfe von Abstraktion (oder Modellierung) ein mathematisches Modell erstellt, dass diesen Körper formal beschreibt. 3D Grafiksoftware, wie Blender [8], wird genutzt um geometrische Körper zu modellieren, texturieren und zu animieren.

- **Prozedurale Modellierung**: *„It encompasses a wide variety of generative techniques that can (semi-)automatically produce a specific type of content based on a set of input parameters.“* [7]
„Prozedurale Modellierung beschreibt generative Techniken, die (semi-)automatisch spezifische, digitale Inhalte anhand von deskriptiven Parametern erzeugen“ (*Übersetzt durch den Autor*)

- **Inverse prozedurale Modellierung (IPM)**:*Aliaga et al.* [1] spricht bei der inversen prozeduralen Modellierung von dem Finden einer prozeduralen Repräsentation von Strukturen bestehender Modelle.

Die Modellierung mithilfe von Grafiksoftware ist eine vergleichbar händische, langwierige Erstellung von Objekten. Hierbei hat der Designer (= Modellierer) die volle Kontrolle über die Strukturen des Objektes.
Bei der prozeduralen Modellierung werden spezifische Strukturen eines zu erstellenden physikalischen Objektes generalisiert und meist über eine Grammatik und globale Parameter abgebildet. Während bei der klassischen Modellierung die menschliche Intuition und bei der prozeduralen Modellierung eine parametrisierte Grammatik vorausgesetzt wird, arbeitet IPM mit bestehenden Modellen und extrahiert („lernt“) die Strukturen des Objektes, die automatisch in eine formale Grammatik überführt werden können. Die Generierung von prozeduralen Modellen ist ein wichtiges, offenes Problem [1].
Aktuelle Ansätze sind:

- Segmentierung von geometrischen Objekten in Ähnlichkeitsgruppen, um Muster (*engl. Patterns*) zu erkennen und

- Kontrollierte Generierung durch Finden optimaler Prameter und Regeln

1.3 Vorangehende Arbeiten

Folgende Stichworte werden später ausführlich erklärt und die Quellen markiert (Version 1.0):

- Prozedurale Modellierung
- Inverse Prozedurale Modellierung
- Formale Grammatik (parametrisiert und probabilistisch)
  - L-Systeme (spezielle, formale Grammatik zur beschreibung von natürlicher Vegetation)
- Neurale Netzwerke zum Erkennen von Objekten und Strukturen

2 Ziele der Arbeit

- Erarbeitung eines Software-/Harwarestacks
- Erstellung eines Projektplans
- Herausarbeiten einer Softwarearchitektur
- Aufsetzen eines „Bachelortagebuchs“
- Prototypische Entwicklung eines Systems zur Erstellung „ähnlicher“ 3D-Objekte anhand eines vorgefertigten Input-Modells mit grafischer Oberfläche
- Schriftliche Ausarbeitung der Ergebnisse

3 Ablauf

- Zweiwöchige Meilensteine mit Besprechungen (Videokonferenz)
- Phasen:
  - Vorbereitung
  - Literaturstudium
  - Problemstudium
  - Praktische Arbeit
  - Schriftliche Arbeit

4 Organisatorisches

- 12. Oktober 2020: Anmeldung der Bachelorarbeit
- Online Repository [3]
- Zweitgutachter: - (bis spätentens 12.10.2020 auswählen)

5 Zeitplan

- ☑ bis 29.08.2020: Erarbeitung Basisquelle
- ☑ 14.09.2020: Kennenlerngespräch, erstes Themengespräch
- ☑ bis 28.09.2020: Version 0.1 des Exposés
- ☐ bis 05.10.2020: Version 1.0 des Exposés, Zweitgutachter auswählen
- ☐ bis 12.10.2020: Literaturrecherche, Anmeldung der Bachelorarbeit
- ☐ bis 16.10.2020: Ausarbeitung des Software-/Hardwarestacks
- ☐ bis 26.10.2020: Erstellung des Projektplans
- ☐ ab 26.10.2020: Bearbeitung des praktischen Teils der Bachelorarbeit, paralleler Beginn des schriftlichen Teils

Abbildungsverzeichnis

Literatur

[1] Daniel G. Aliaga, Ilke Demir, Bedrich Benes, and Michael Wand. Inverse procedural modeling of 3d models for virtual worlds. In *ACM SIGGRAPH 2016 Courses*, SIGGRAPH '16, New York, NY, USA, 2016. Association for Computing Machinery.

[2] Cheloniidae 3. `http://spencertipping.com/cheloniidae/`. Turtle graphics library for Java. Accessed 29.09.2020.

[3] Github. `https://github.com/adrian-helberg/bachelor`. GitHub, Inc. (2020).

[4] Jianwei Guo, Haiyong Jiang, Bedrich Benes, Oliver Deussen, Xiaopeng Zhang, Dani Lischinski, and Hui Huang. Inverse procedural modeling of branching structures by inferring l-systems. *ACM Trans. Graph.*, 39(5), June 2020.

[5] P Prusinkiewicz. Graphical applications of l-systems. In *Proceedings on Graphics Interface '86/Vision Interface '86*, page 247–253, CAN, 1986. Canadian Information Processing Society.

[6] Kripasindhu Sarkar, Kiran Varanasi, and Didier Stricker. Trained 3d models for cnn based object recognition. In *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 130–137, 01 2017.

[7] Ruben M. Smelik, Tim Tutenel, Rafael Bidarra, and Bedrich Benes. A survey on procedural modelling for virtual worlds. *Comput. Graph. Forum*, 33(6):31–50, September 2014.

[8] Pablo Vazquez and Francesco Siddi. About blender. `https://www.blender.org/about/`, 2017. Blender Foundation (2002).