KLASSIFIKATOR ZUM EVALUIEREN VON TETRAEDER-TOPOLOGIEN IM KONTEXT GESCHLOSSENZELLIGE 3D-DRUCK-STRUKTUREN AUF BASIS EINES KÜNSTLICHEN NEURONALEN NETZES

Alexander Prinz

ein privates Projekt als Teilprojekt

zum Optimieren der Festigkeit von 3D-Druck-Teilen

Das Projekt befasst sich mit der Validierung von Tetraeder-Topologien zur Erkennung von möglichen Fehlern bei der algorithmischen Erzeugung von semi-stochastisch erzeugten Strukturen im Kontext 3D-Druck. Es handelt sich um ein privates Projekt zur Festigkeitsoptimierung von porösen 3D-Druckstrukturen.

Ziel ist via Delaunay-Triangulation aus einer semi-stochastisch erzeugten Punktwolke eine 3D-druckbare Struktur zu erzeugen. Die folgende Abbildung zeugt ein 3D-Modell einer solchen Struktur (siehe Abb. I).

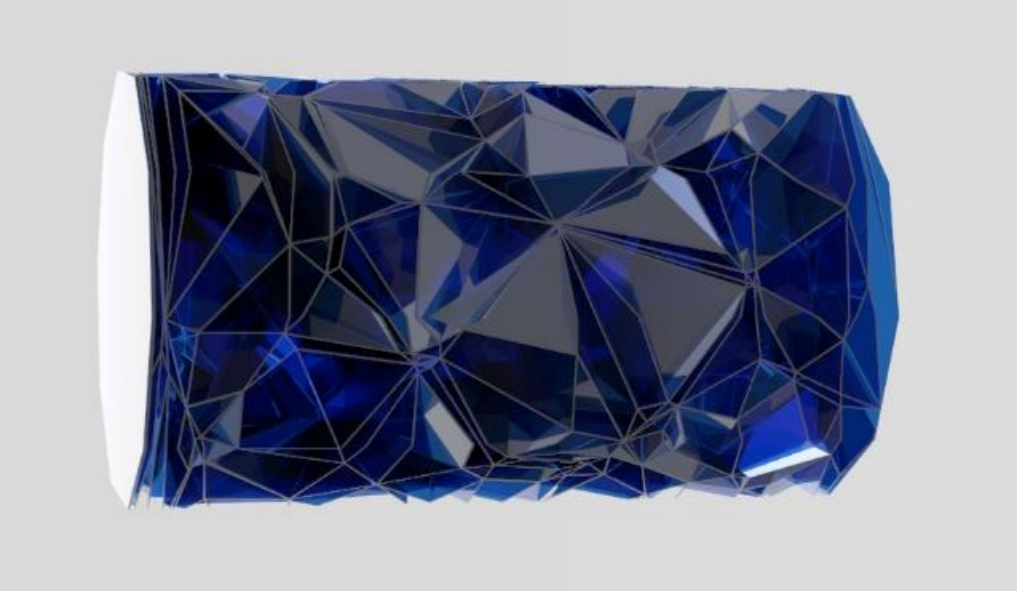

Abb.: I

Die Abbildung zeigt ein 3D-Modell der Tetraedrischen Struktur.

Bei der Delaunay-Triangulation werden die Punkte der Punktwolke trianguliert. Es sind somit Punkt-Beziehungen in Form von Kanten bekannt. Damit aber eine reale Struktur gedruckt bzw. erst ermöglicht werden kann, müssen aus den Tetraedern 3-dimensionale Wände aus den Facetten berechnet werden. Dazu wurde von mir ein Algorithmus entwickelt, welcher jede der 4 Flächen (Facetten) eines Tetraeders in den Tetraeder um eine bestimmte Distanz verschiebt. Die Schnittpunkten dieser so verschobenen Flächen beschreiben einen neuen Tetraeder (Tochtertetraeder), welcher etwas kleiner ist, als sein Ursprungstetraeder (Muttertetraeder). Betrachtet man den Muttertetraeder als ein reales Volumen und zieht von diesem das Volumen des berechneten Tochtertetraeder ab, so erhält man ein Hohlkörper, welcher real Druckbar ist. Als Konglomerat vieler Tetraeder entsteht so eine geschlossenzellige Struktur, wie sie in Abbildung 1 zu sehen ist.

Unter bestimmten Umständen bzw. ungünstigen Geometrieparametern, versagt der Algorithmus zum Erzeugen der Tochtertetraeder jedoch. Folgende Abbildungen sollen dies verdeutlichen. In Abbildung 2 sieht man, mit schwarzen Kanten und Eckpunkten dargestellt, den Muttertetraeder. In grün den zugehörigen Tochtertetraeder. Jedoch ist die topologische Beziehung beider Tetraeder valide, da sich keine ihrer Facetten schneiden. Mit einem solchen Tetraederpaar ist eine valide 3D-druckbare Struktur

erzeugbar. In Abbildung 3 sieht man dagegen ein Tetraederpaar, welches eine invalide Topologie aufweist, da sich einige ihrer Seiten schneiden.

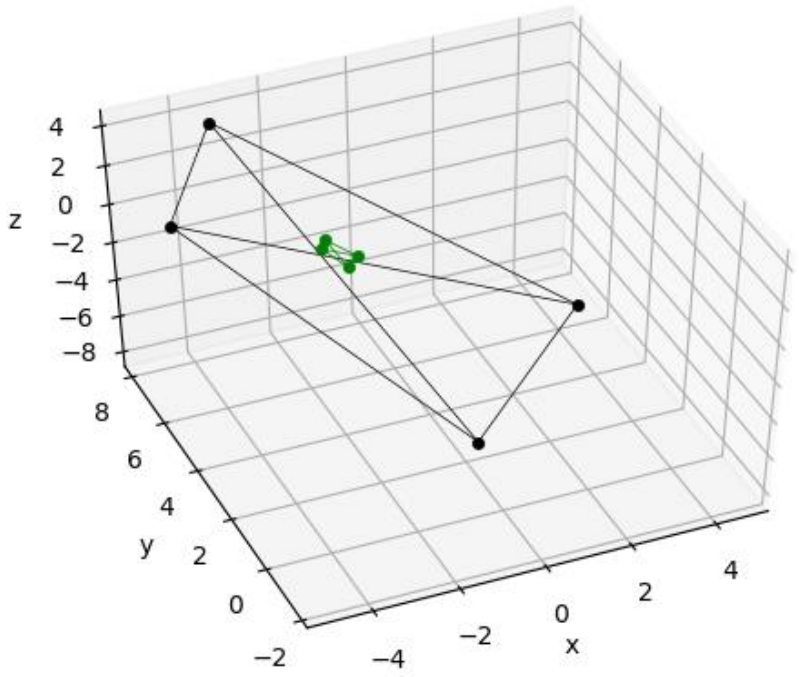

Abb.: II

Tetraederpaar (Muttertetraeder in schwarz, Tochtertetraeder in grün) mit valider Topologie d.h. keine Facetten (Tetraederflächen) schneiden einander.

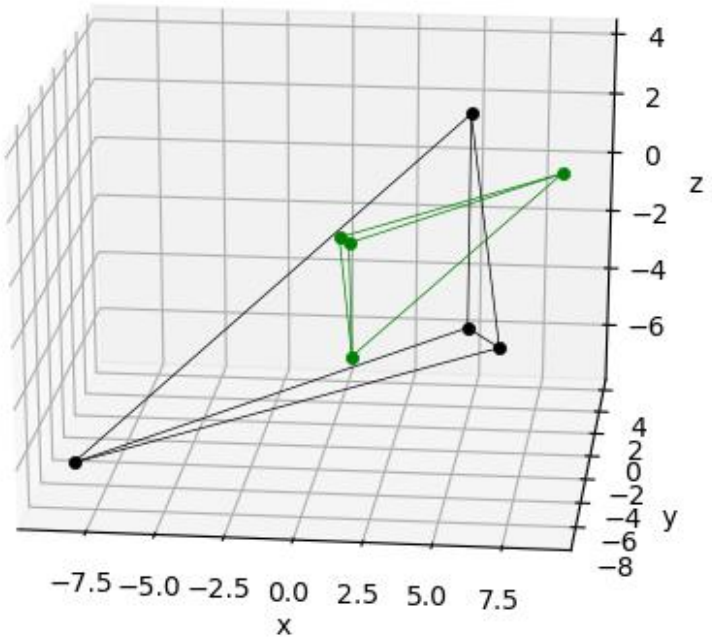

Abb.: III

Tetraederpaar (Muttertetraeder in schwarz, Tochtertetraeder in grün) mit invalider Topologie d.h. einige Facetten (Tetraederflächen) schneiden einander.Zu sehen an der annähernd zur y-z-Ebene parallelen Facette des Muttertetraeders, welche durch das Tochtertetraeder durchschnitten wird.

Der Algorithmus, welcher das Tochtertetraeder berechnet, berechnet genauer gesprochen die 4 aufspannenden Punkte.

Die Idee ist nun, einen Klassifikator zu entwickeln, welcher in der Lage ist, sowohl anhand der 4 Punktkoordinaten des Muttertetraeders, sowie auch denen des Tochtertetraeders, zu klassifizieren, ob die Topologie beider valide ist oder nicht.

Dazu wurde ein künstliches neuronales Netz entwickelt, welches das Klassifizierungsproblem lösen soll.

Es besteht aus 24 Eingangsneuronen, wobei die ersten 12 die Punktkoordinaten des Muttertetraeders entgegennehmen. Die weiteren 12 Eingangsneuronen nehmen entsprechend die des Tochtertetraeders entgegen. Zudem kommt ein Hidden Layer mit 128 Neuronen zum Einsatz und mündet in einem einzigen Ausgangsneuron. Zwischen Hidden Layer und Ausgangsneuron ist eine Sigmoid-Aktivierungsfunktion geschalten. Da der Klassifikator lediglich binär entscheiden muss, also ob die Topologie valide ist oder nicht. Dabei soll der Wert 1 Validität repräsentieren und der Wert 0 Invalidität.

Als Trainingsdaten dienen randomisiert Erzeugte Muttertetraederpunkte und durch den Algorithmus erzeugte Tochtertetraederpunkte sowie das zugehörige Label (1 oder 0). Die Trainingsdaten wurden via Visualisierung eigenständig „per Hand“ gelabelt.

Die Trainingsdaten wurden so als CSV-Dateien gespeichert.

Um ausbalancierte Trainingsdaten zu haben, wurde ein Skript (data_balancer.py) entwickelt, welches die Daten hinsichtlich der Label ausbalanciert. D.h. die Trainingsdaten enthalten genau soviel Fälle valider Tetraeder-Topologien wie auch invalider.

Zudem wurden zur Modelvalidierung auf gleicher weise ausbalancierte Validierungsdaten erzeugt. Diese sind unabhängig von den Trainingsdaten.

Die genaue Parametrisierung des neuronalen Netzes soll hier nicht genauer erläutert werden. Dazu kann das entsprechende Python-Skript, welches zum trainieren des Netzes verwendet wurde (NN.py) eingesehen werden.

Folgende Abbildung (Abb. IV) zeugt die Verlusstfunktionswerte sowohl bzgl. der eigentlichen Trainingsdaten als auch der der Testdaten über den gesamten Optimierungsprozess.

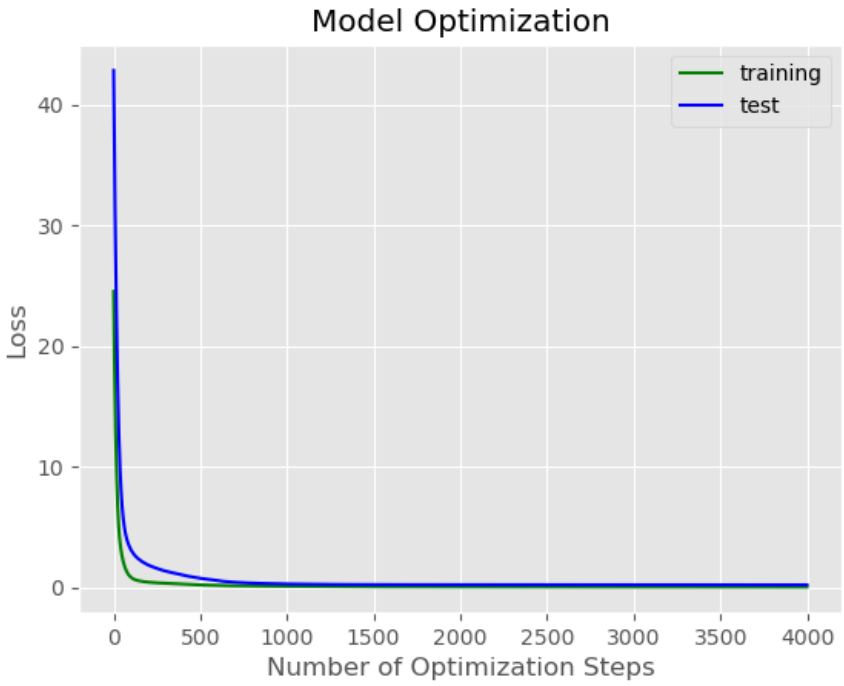

Abb.: IV

Visualisierung der Optimierungsprozesses hinsichtlich der Verlustfunktionswerte über den Modeloptimierungsprozess.

Nach dem Training wurde das neuronale Netz mit den Validierungsdaten auf seine Genauigkeit validiert. Dazu wurde das Python-Skript NN_validation.py entwickelt.

Das Modell hat eine Genauigkeit von 0.9571.

Das Modell dient also sehr gut als Klassifikator zum Evaluieren der Tetraeder-Topologie.