**Softwaremanagement Bachelorabschlussarbeit**

**THEMA DES PROJEKTES:**

Implementierung von spröder und duktiler Deformierung und Fraktur von Mesh-Components in der Unity-Engine

**ZIELE**:

* Die Software soll als ein ECS-Component (als Skript) der Unity-Engine entwickelt werden, welches GameObjects erweitert.
* Das System soll verschiedene physische Materialien bereitstellen, die bestimmen, wie sich das GameObject bei Deformierung verhält (z.B. Glas).
* Die Genauigkeit / Feinkörnigkeit des zugrundeliegenden Finite-Elemente-Modells muss vom Entwickler konfigurierbar sein.
* Das System soll auf Kollisionen und die dadurch entstehenden Kräfte reagieren können.
* Das System soll auf Deformierungen und die dadurch entstehenden Kräfte reagieren können.
* Das System soll selbst unter größeren Deformierungen / Frakturen performant bleiben.
* Das Mesh des GameObjects muss bei Deformierungen und Frakturen entsprechend aktualisiert werden, sodass korrektes Rendering / Shading stattfinden kann.
* Falls eine Teilung des Objektes entsteht, sodass es in zwei oder mehrere Teile zerbricht, sollen die einzelnen Teile als neue GameObjects mit einem eigenen Mesh, einem eigenen Collider und dem System als Skript-Component als Kinder des ursprünglichen GameObjects eingetragen werden. Das Elternelement soll sein Mesh-Component, seinen Collider und das System als Skript-Component entfernen.

**VERWENDUNGSSZENARIEN:**

Beispiele von simulierbaren Objekten:

* Eine statische Wand aus Holz / Beton / Backsteinen
* Ein Hausrenovierungsgerüst
* Schwerter
* Porzellanschale
* Löffel
* Pipeline
* Fenster
* Holzzaun

Beispiele von Kräften:

* Kollision
* Streckung
* Stauchung
* Biegung
* Verdrehung

Die simulierbaren Objekte sind mit allen Kräften beliebig kombinierbar, sodass sich schnell eine Reihe von möglichen Verwendungszwecken formulieren lassen.

**VORGEHENSMODELL:** SCRUM / inkrementell

**MOMENTANER STAND DES PROJEKTES:**

* Das Kraftanalysesystem ist implementiert und das Remeshingsystem befindet sich in der Debuggingphase.
* Es fehlt noch eine Umsetzung eines robusten Diskretisierungssystems, das konfigurierbar ist.
* Es fehlt ein System, dass erkennt, wenn Teilungen stattfinden und entsprechend Kind-GameObjects instanziiert.
* Es fehlt ein Algorithmus, welcher das Datenmesh des Kraftanalysesystems in ein Mesh-Component umfiltert und außerdem einen Collider dafür generiert.

**WIE SOLL ES WEITERGEHEN:**

* Momentan werden Testfälle für das Remeshing umgesetzt, das bereits implementiert, aber noch fehlerhaft ist. Diese Testfälle sollen erweitert werden, sodass ein robustes System entsteht.
* Es sollen weiterhin Testfälle für das Kraftanalysesystem entwickelt werden, die die Robustheit des Systems zusichern.
* Sobald die beiden Systeme (Kraftanalyse und Remeshing) robust genug sind, geht es darum, die bisher ausschließlich spröde Kraftanalyse auf duktil zu erweitern und dieses entsprechend gut zu testen.
* Da im Moment ausschließlich Quader diskretisiert werden können, muss daraufhin ein Algorithmus entwickelt werden, der alle möglichen Meshes diskretisieren kann.
* Für verschiedenste GameObjects sollen verschiedenste Testsituationen entworfen werden (s. Liste von Kräften).
* Das Datenmesh muss entsprechend in einen MeshComponent eingepflegt werden.

**ANFORDERUNGEN:**

Subsystem – FEM-Diskretisierung:

* Das System muss jedes Unity-GameObject, an das es angehängt wird, in Tetraeder diskretisieren können.
* Der benutzende Entwickler muss über Parameter des Skriptes einstellen können, wie hoch die Auflösung der Diskretisierung ist.
* Die Auflösung muss in alle 3 Dimensionen konfigurierbar sein.
* Die entstehende Datenstruktur muss in einer Datei speicherbar sein.
* Die entstehende Datenstruktur muss aus einer Datei lesbar sein.
* Das zu diskretisierende Objekt muss die Information verwalten, ob bereits eine Datei mit den Diskretisierungsdaten existiert, oder nicht.
  + Falls die Datei nicht vorhanden ist, muss sie erstellt und mit den Diskretisierungsdaten befüllt werden.
  + Falls die Datei vorhanden ist, muss sie geladen werden können, sodass die Diskretisierungsdaten für das Skript verfügbar sind.

Subsystem – interne Kraftanalyse (Deformierung):

* Das System muss die durch Deformierung entstehenden internen Kräfte erkennen können und dem Frakturerkennungssystem zur Verfügung stellen.

Subsystem – externe Kraftanalyse (Kollisionen):

* Das System muss die durch Kollisionen entstehenden externen Kräfte erkennen können und die dadurch entstehenden internen Kräfte dem Frakturerkennungssystem zur Verfügung stellen.

Subsystem – Frakturerkennung:

* Die Frakturerkennung muss erkennen, wann eine Fraktur auftritt und muss dem Remeshingsystem kommunizieren können, wie die Fraktur voranschreitet.

Subsystem – Remeshing:

* Das Remeshingsystem muss die Datenstruktur entsprechend der Frakturdaten des Frakturerkennungssystems aktualisieren.
* Das Remeshingsystem ist außerdem dafür zuständig, zu erkennen, wann eine Fraktur eine Teilung hervorruft und muss dies entsprechend an die ECS-Anpassung kommunizieren.

Subsystem – ECS-Anpassung:

* Die ECS-Anpassung muss die Veränderungen der Datenstruktur (durch Verformung oder Fraktur) an das Mesh-Component des GameObjects kommunizieren.
* Weiterhin muss die ECS-Anpassung eine eventuelle Teilung des GameObjects managen, indem sie Kindobjekte erstellt, die die Teilmeshes enthalten und wiederum auch Collider und das Frakturskript besitzen.

**ZEITPLAN:**

Ich muss das Kolloquium bis zum 31.08.2020 gehalten haben. Das bedeutet, dass ich die Software und die Ausarbeitung bis zum 17.08.2020 abgegeben haben muss. Das wiederum bedeutet, dass ich maximal noch 9 ½ Wochen habe, um meine Software und Ausarbeitung abzuschließen.

Ich rechne mit mindestens 2 Wochen für die Bearbeitung der Ausarbeitung.

Da ich SCRUM verwenden will, werde ich die Projektzeit in Sprints unterteilen.

s. Schedule.xlsx

Meilensteine:

* Aufwandsabschätzung für die nächsten Meilensteine abgeschlossen
* Remeshing (inklusive Teilungserkennung) funktioniert
* Robustes Testen von interner Kraftanalyse und Remeshing erfolgreich
* Externe Kraftanalyse funktioniert & ist robust getestet
* ECS-Anpassungssystem funktioniert & ist robust getestet
* Grundlegende Struktur der Ausarbeitung steht fest und besitzt inhaltliche Stichpunkte
* Ausarbeitung ist auf 40 Seiten ausformuliert
* Ausarbeitung wurde finalisiert

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | Remeshing |
|  |  |  |  |  |  | Testen von  internen Kräften und Remeshing |
|  |  |  |  |  |  | Externe Kraftanalyse |
|  |  |  |  |  |  | ECS-Anpassung |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  | 40 Seiten Ausarbeitung |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  | Software und Ausarbeitung finalisiert |
|  |  |  |  |  |  |  |

