Tool - & Pluginprogrammierung - Hausarbeit

von Janina Althöfer

Einleitung

Für die Animation von Menschen oder anderen Lebewesen in Spielen werden die Bewegungen mittels Motion Capture definiert. Diese Motion Capture werden von Schauspielern in einem speziellen Anzug aufgenommen, an dem Tracker angebracht sind, welche die Position und Rotation des Punktes aufzeichnen.

Die aufgezeichneten Daten werden dann in einer Datei gespeichert. Eines der Formate für Motion Capture Dateien ist das BioVision Hierarchy (BVH) Format. Dieses Format ist auf eine bestimmte Weise aufgebaut und ermöglicht ein einfaches rekursives Einlesen der Trackerpositionen sowie der Bewegungsdaten.

Im Rahmen der Hausarbeit wurde ein Tool zur Anzeige der Motion Capture Daten und die Animation der Bewegung implementiert.

Implementierung

Das Dateiformat ist in zwei Abschnitte unterteilt. Im ersten Teil wird die Hierarchie der eingelesenen Tracker angegeben.
Dabei gibt es zuerst einen Wurzelknoten, von dem alle anderen Tracker relativ positioniert sind. Dieser Knoten wird durch das Schlüsselwort ROOT mit anschließendem Namen der Trackerposition angegeben.
Alle weiteren Tracker und ihre Positionen haben entweder das Schlüsselwort JOINT oder End Site. Ein Joint ist dabei eine vollständige Angabe der Positionen sowie der Rotation relativ zu dem Vorgänger.
Das Schlüsselwort End Site dagegen wird nur die Position aufgelistet, da es sich um das Ende eines rekursiven Pfads handelt.

In jedem Joint sind angegeben wie sich die Position zu dem Elternknoten verändert wie viele Kanäle für Position und Rotation in den Bewegungsdaten vorliegen und in welcher Reihenfolge diese angegeben sind. Außerdem wird gespeichert, an welcher Stelle der Bewegungsdaten die einzelnen Werte zu finden sind.

Während des Ladens der Hierarchiedaten wird solange kein Ende eines Rekursionspfads erkannt wird, die jeweiligen Kinder des aktuellen Knoten in einem Vektor gespeichert. Außerdem wird der Elternknoten für jeden Knoten gespeichert.
Somit ergeben sich drei Arten von Joints: der Wurzelknoten, Pfadknoten und Endknoten.

Ist das Laden der Hierarchie beendet, werden die Bewegungsdaten geladen.
Diese bestehen zum einen aus der Anzahl der aufgezeichneten Frames und der Frame Rate, welche bestimmt wie viele Frames pro Sekunde gezeichnet werden müssen und zum Anderen aus den eigentlichen Bewegungsdaten, welche dann in ein zweidimensionales Array geladen werden.

In den BVH – Datein die zur Implementierung vorlagen, war die Anzahl der Frames sehr hoch, sodass es eine gewisse Zeit dauert bis das Laden der Daten beendet ist.

Dies konnte auch durch ein Umstrukturieren in ein eindimensionales Array nicht verändert werden und wurde daher so belassen.

Sind die kompletten Daten geladen, so wird die sich aus den Daten ergebene Figur gezeichnet.
Das Rendern erfolgt dabei mit den alten OpenGL Objekten und des Matrix Stack, da sich diese für eine rekursive Berechnung der Transformationsmatrizen sehr gut eignen.
Insbesondere der Matrix Stack, mit einer begrenzten Tiefe von 32 Matrizen könnte dabei als problematisch erachtet werden, es ist allerdings eher selten, dass ein einziger Pfad der Figur über 32 Matrizen verbraucht.
Dementsprechend wurde der Stack verwendet, da ansonsten ein eigener Matrix Stack implementiert worden wäre. Das doppelte Implementieren wäre dabei unisinnig gewesen.

Sollte sich im Laufe der Zeit herausstellen, dass der Matrix Stack keine ausreichende Größe hat, kann die Berechnung der jeweiligen Matrizen von Hand erfolgen, da dafür die Joints nur um eine Matrixrepräsentation erweitert werden müssen. Dies wäre notwendig, da die relativen Matrizen der einzelnen Joints rekursiv mit den Matrizen ihrer Eltern multipliziert werden müssen. Dies geschieht momentan automatisch durch Abfragen der entsprechenden Werte aus dem Array.
Dabei wird immer der aktuelle Frame übergeben, sodass die aktuellen Daten bekannt sind und so die entsprechende Stelle des Arrays ausgelesen werden können.
Diese werden dann durch Translations- bzw. Rotationsmatrizen an die aktuelle Matrix anmultipliziert.

Für die Länge der einzelnen “Knochen” zwischen den Trackerpositionen werden dann durch Zylinder realisiert, welche in OpenGL zur Verfügung stehen.
Diese Zylinder sind heutzutage auch eher unüblich, allerdings war es nicht Teil des Tools eine besonders schöne Darstellung der Figur zu bekommen, sodass die Zylinder ausreichen.
So wurde auch das Implementieren von Vertices und Shadern übergangen.

Schlussendlich wurde das Tool so implementiert, dass es am Ende einer Animation wieder am Anfang der Animation beginnt.
Es ist auch möglich die Animation zu Pausieren um sich einen bestimmten Frame in Ruhe anzuschauen. dabei können die vorherigen und nächsten Frames durch Tastendruck angezeigt werden.

Die Anzeige ist insofern dynamisch, als dass man die Position der Kamera (Distanz, Höhe und Drehwinkel) beeinflussen kann, um die Animation von allen Seiten zu betrachten.

Außerdem wurde eine simple Windowsdialogbox implementiert, sodass man während des Laufens des Programms die geladene Datei ändern kann.

Bedienung des MoCap Viewer

W	Kamerahöhe erhöhen
S	Kamerahöhe verringern
A	um den LookAt Punkt rotieren (im Uhrzeigersinn)
D	um den LookAt Punkt rotieren (gegen den Uhrzeigersinn)
Q	Kameraposition heran bewegen
E	Kameraposition zurück bewegen
L	Laden einer BVH – Datei
P	Pausieren des Abspielens / der Animation
I	Vorheriger Frame (wenn pausiert)
O	Nächster Frame (wenn pausiert)