Visualisierung hierarchisch aufgebauter Bewegungsdaten

Carl Bittendorf[1]

[1]Karlsruher Institut für Technologie, Karlsruhe

E-Mail: carl.bittendorf@gmail.com

Schlüsselwörter: Motion Capture, BVH, Gradientenabstieg

Einleitung

Videobetrachtung ist im Leistungssport ein wichtiges technisches Hilfsmittel zur Informationspräsentation und -rückmeldung. Dabei kann die Aufmerksamkeit von Athleten gezielt auf Details der Bewegung gerichtet und die Bewegungsvorstellung der Sportler mit einer objektiven Außenperspektive verglichen werden (Mayer & Hermann, 2009). Anstatt herkömmlichen Videoaufzeichnungen kann Motion Capture eingesetzt werden, was die Betrachtung aus einer beliebigen Perspektive ermöglicht. Außerdem können Bewegungen so nachträglich bearbeitet und damit Technikfehler korrigiert werden. Allerdings ist die Darstellung bisher meist auf einfache Strichfiguren beschränkt.

Dem Institut für Angewandte Trainingswissenschaft liegen Daten zu den Bewegungen von Spitzensportlern im hierarchisch aufgebauten Dateiformat BioVision Hierarchy (BVH) vor (siehe Meredith & Maddock, 2001). Dessen Aufbau ist baumartig, wobei ausgehend von einem einzelnen Wurzel-Gelenk jedes Gelenk eine beliebige Anzahl an untergeordneten Gelenken besitzen kann. Dabei sind die Positionen einzelner Gelenkpunkte nicht über globale Koordinaten, sondern rekursiv mithilfe von Abständen (Offsets) und Rotationen relativ zu ihren übergeordneten Gelenken definiert.

Das Ziel ist die Erstellung von realistischen Bewegungsanimationen auf Basis dieser BVH-Dateien. Entsprechende Animationsprogramme benötigen aber einen bestimmten hierarchischen Aufbau, der sich von dem der vorliegenden Daten unterscheidet.

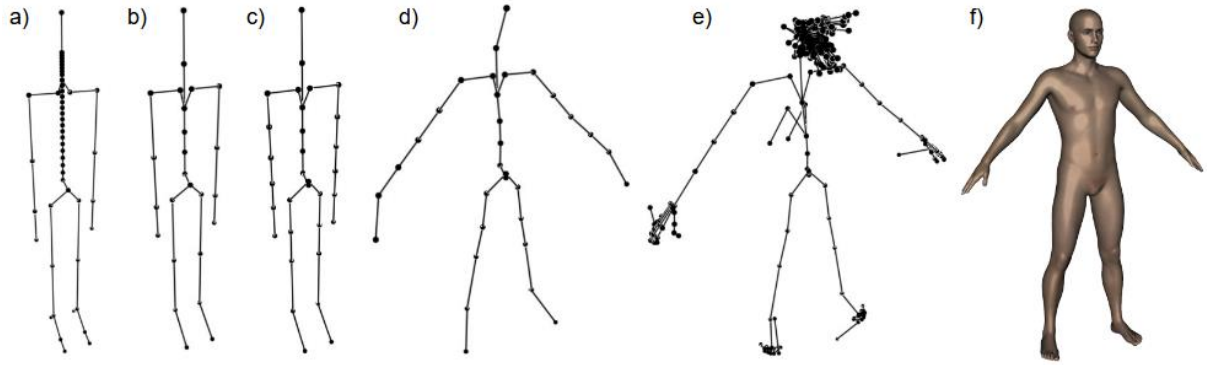

Abb. 1. Beispielhafter Transformationsprozess. a) Ausgangs-Hierarchie der Software alaska/Dynamicus b) Entfernen von Gelenken c) Hinzufügen von Gelenken d) Übertragen der Offsets von der Zielhierarchie e) Übertragen der Rotationen auf die Zielhierarchie f) Volumenkörper im Animationsprogramm DAZ3D.

In Abbildung 1 ist zu sehen, dass beispielsweise einige Gelenke der Ausgangsdaten (a) keine Entsprechung in der Zielhierarchie (e) haben und umgekehrt. Zur Lösung dieses Problems wurden Algorithmen entwickelt, mit denen eine Veränderung von Gelenk-Hierarchien unter Beibehaltung der ursprünglichen Bewegung möglich ist.

Diese Arbeit fokussiert sich auf ein zentrales Problem dabei, nämlich die Anpassung der Offsets und Rotationen nach dem Entfernen von Gelenkpunkten. In einer BVH-Datei ist der Offset zwischen zwei benachbarten Gelenken konstant, die Rotationen eines Gelenks werden für jedes Bild in der Bewegungssequenz durch jeweils drei Euler-Winkel definiert. Zusätzlich besitzt das Wurzel-Gelenk für jedes Bild Positionsangaben, mit denen eine Translation der kompletten Gelenk-Hierarchie möglich ist. Die globale Position

$$\hat{\boldsymbol{p}}_{f,v} = \hat{\boldsymbol{p}}_{f,v-1} + \left(\prod_{i=1}^{v-1} R_{f,i} \right) \boldsymbol{o}_v$$

eines Gelenks v für ein bestimmtes Bild f ergibt sich aus der globalen Position $\hat{\boldsymbol{p}}_{f,v-1}$ des übergeordneten Gelenks sowie den Rotationen R der Gelenkpunkte des Pfades vom Wurzel-Gelenk zum v übergeordneten Gelenk multipliziert mit dem Offset $\boldsymbol{o}_v$ von v. Beim Wurzel-Gelenk ergibt sich die globale Position lediglich aus dem Offset sowie den Positionsangaben. Änderungen der Rotation eines Gelenkpunktes wirken sich in einem hierarchischen Dateiformat automatisch auf die globalen Positionen der untergeordneten Gelenke aus, wobei die Abstände zwischen benachbarten Punkten konstant bleiben, wie es von benachbarten Gelenken des menschlichen Körpers zu erwarten ist. Dies trifft aber in der Regel nicht auf Gelenkpunkte zu, die ursprünglich nicht benachbart waren, sodass es meist unmöglich ist, für jedes Gelenk und jedes Bild die Original-Positionen zu erreichen. Die resultierenden Positionen sollten allerdings den Original-Positionen möglichst gut entsprechen.
Es wird angenommen, dass die entwickelten Algorithmen die aus einer Gelenk-Hierarchie resultierenden Abweichungen von den Original-Positionen durch Anpassung der Offsets und Rotationen minimieren können.

Methode

Zunächst wird eine Verlustfunktion benötigt, die das Ausmaß der Abweichungen der derzeitigen Positionen vom Original als eine einzelne positive Zahl wiedergibt. Je größer die Zahl desto größer die Unterschiede. Damit lässt sich feststellen, welche Offsets und Euler-Winkel die Original-Bewegung am besten repräsentieren. Im Folgenden wird die Summe der Abweichungsquadrate

$$L_f = \sum_{v=1}^{V} \left| \boldsymbol{p}_{f,v} - \hat{\boldsymbol{p}}_{f,v} \right|^2$$

verwendet, wobei $\boldsymbol{p}_{f,v}$ die Original-Positionen für das entsprechende Bild und V die Gelenke sind. Nützliche Eigenschaften der Summe der Abweichungsquadrate sind Symmetrie, Differenzierbarkeit und leichte Interpretierbarkeit. Außerdem werden durch das Quadrieren größere Differenzen stärker gewichtet als kleinere, wodurch besonders große Abweichungen eher vermieden werden können.
Für die Minimierung der Abweichungen wurden zwei Algorithmen entwickelt, einer für die Offsets und einer für die Rotationen. Da die Auswirkungen der Rotationen abhängig von den Offsets sind, müssen die Offsets zuerst angepasst werden. Könnten die Längen der Offsets für jedes Bild individuell gewählt werden, so wäre es möglich, für jedes Bild alle Gelenke in ihre Originalpositionen zu bringen. Da dies aber nicht möglich ist, muss für jeden Offset ein Betrag gefunden werden, der die Summe der quadratischen Abweichungen zu den theoretisch optimalen Längen minimiert. Diese Eigenschaft erfüllt das arithmetische Mittel

$$\bar{o}_v = \frac{1}{F} \sum_{f=1}^{F} \left| \boldsymbol{p}_{f,v} - \boldsymbol{p}_{f,v-1} \right| .$$

Als Algorithmus formuliert wird für jedes Gelenk v, falls es nicht das Wurzel-Gelenks ist, der Offset $\boldsymbol{o}_v$ mit dem Quotienten aus dem Mittelwert und dem derzeitigen Betrag multipliziert. Dadurch bleibt die Richtung des Offsets erhalten.

Da die Rotation eines Gelenks die Positionen aller untergeordneten Gelenke beeinflusst, müssen alle zu optimierenden Rotationen eines Bildes simultan behandelt werden. Die Gleichungen, welche die globalen Positionen der einzelnen Gelenke ergeben, müssen als eine einzelne Funktion aufgefasst werden, deren Differenzen zu den Original-Positionen es zu minimieren gilt. Die zu optimierenden Parameter sind die Euler-Winkel. Der folgende Algorithmus basiert auf dem Gradientenabstieg, einem iterativen Optimierungsalgorithmus zum Finden eines lokalen Minimums einer differenzierbaren Funktion. Dabei werden wiederholt Schritte in die entgegengesetzte Richtung des Gradienten der Funktion am derzeitigen Punkt unternommen.

Der Algorithmus 1) berechnet den Verlust $L_f(\theta_t)$ mit den aktuellen Parametern θ_t , 2) die neuen Parameter θ_{t+1} ergeben sich aus der Differenz zwischen den vorherigen Parametern und dem Gradienten der Verlustfunktion multipliziert mit einer Schrittlänge (Lernrate) η:

$$\theta_{t+1} = \theta_t - \eta \nabla L_f(\theta_t).$$

Der Subtrahend wird auch Aktualisierungsvektor genannt. Es folgt die 3) beliebige Wiederholung der Schritte 1 und 2. Anschließend werden 4) die Parameter mit dem geringsten Verlust aller Wiederholungen gespeichert. Die Schritte 1 bis 4 werden für jedes Bild ausgeführt.

Wiederholungszahl und Lernrate müssen vom Benutzer festgelegt werden, wobei die einzigen Nachteile einer höheren Wiederholungszahl Zeit und Rechenaufwand sind. Ist die Lernrate zu groß, konvergiert der Algorithmus nicht Schritt für Schritt auf ein lokales Minimum, stattdessen vergrößert sich der Verlust sogar. Ist die Lernrate allerdings zu klein, sind entweder sehr viele Wiederholungen nötig oder der Algorithmus bleibt in einem suboptimalen lokalen Minimum „stecken“.

Der Algorithmus bietet zahlreiche Anpassungs- und Erweiterungsmöglichkeiten. So ist es möglich, ihn lokal einzusetzen. Wurden beispielsweise nur Gelenke des Oberkörpers entfernt, können die Gelenke des Unterkörpers von der Optimierung ausgeschlossen werden, da sie bereits optimale Rotationen besitzen. Im Bereich des maschinellen Lernens wurden Variationen in der Berechnung des Aktualisierungsvektors (Optimizer) entwickelt. Der Optimizer Adaptive Momentschätzung (ADAM) berechnet beispielsweise adaptive Lernraten für jeden Parameter (Kingma & Ba, 2014) und lässt den Algorithmus somit in der Regel schneller und zuverlässiger konvergieren. Die Benutzung von Startparametern mit einem möglichst kleinen Verlust kann die Resultate ebenfalls verbessern und ermöglicht die Verwendung kleinerer Lernraten. Dafür können entweder Heuristiken eingesetzt oder die optimierten Parameter des jeweils vorherigen Bildes in einer Bewegungssequenz herangezogen werden. Darüber hinaus ist das Steuern der Lernrate abhängig von der Wiederholungszahl oder das Anpassen von Wiederholungszahl und Lernrate abhängig vom Verlust möglich. So könnten beispielsweise Bilder, die nach einer bestimmten

Wiederholungszahl einen relativ hohen Verlust aufweisen, gezielt mit zusätzlichen Wiederholungen optimiert werden.
Die entwickelten Algorithmen wurden in Julia (Bezanson, Edelman, Karpinski & Shah, 2017) implementiert und zur Überprüfung auf zwei BVH-Dateien angewendet, bei denen 24 von 49 Gelenken entfernt wurden (siehe Abbildung 1). Dabei wurde der Optimizer ADAM mit einer Lernrate von 0,1 und 500 Wiederholungen verwendet und jeweils die Summe der Verluste aller Bilder vor und nach der Anwendung verglichen.

Ergebnisse

Der Verlust wurde auf 0,11% bzw. 0,23% des Ausgangsverlusts reduziert. Dadurch kann die ursprüngliche Bewegung mit minimalen Abweichungen während des Transformationsprozesses beibehalten werden.

Diskussion

Die vorgestellten Algorithmen können Abweichungen von den Original-Positionen deutlich reduzieren, wodurch Transformationen von Gelenk-Hierarchien und damit die Erzeugung realistischer Bewegungsanimationen möglich sind. Die Erzeugung von Volumenkörpern auf Basis von Bewegungen ist allerdings auch für andere Zwecke nützlich. Beispielsweise erwähnen Kirmizibayrak, Honorio, Jiang, Mark und Hahn (2011) die Möglichkeit von Athlet-spezifischen Strömungssimulationen im Schwimmen.
Ein Nachteil des Gradientenabstiegs ist sein hoher Zeitbedarf bei vielen Bildern, insbesondere wenn Athleten schnell Feedback erhalten sollen. Dies könnte durch Parallelisierung gelöst werden.
Es existieren bereits Julia-Skripte für die BVH-Hierarchien mehrerer Ausgangsprogramme, die den kompletten Transformationsprozess automatisch durchführen. Die Benutzung und ggf. Anpassung dieser Skripte könnte jedoch zusätzlich durch eine eigenständige Anwendung mit graphischer Benutzeroberfläche erleichtert werden, sodass die Skripte beispielsweise für Trainer zugänglicher sind.

Literatur

Bezanson, J., Edelman, A., Karpinski, S. & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM review, 59* (1), 65–98.

Kingma, D. P. & Ba, J. (2014). Adam: a method for stochastic optimization (2014). *arXiv preprint arXiv:1412.6980, 180*.

Kirmizibayrak, C., Honorio, J., Jiang, X., Mark, R., & Hahn, J. K. (2011). Digital analysis and visualization of swimming motion. *International Journal of Virtual Reality*, *10* (3), 9-16.

Mayer, J. & Hermann, H.-D. (2009). *Mentales Training.* Springer.

Meredith, M. & Maddock, S. (2001). Motion capture file formats explained. *Department of Computer Science, University of Sheffield, 211*, 241–244.