Transformationen

Kurz & Knapp

- Gilt für homogene Transformationsmatrizen Assoziativität?
- Gilt für homogene Transformationsmatrizen Kommutativität?
- Was ist das neutrale Element der Multiplikation von homogenen Transformationsmatrizen?
- Wie verhält sich eine Scherung für ein vom Ursprung etwas entferntes Quadrat?
- Was bedeutet es wenn in einem homogenen Vektor der letzte Eintrag 0 oder 1 ist?
- Was sind die Vorteile bei der Verwendung von Eulerwinkeln?
- Was sind die Vorteile bei der Verwendung von Quaternionen?

Aufgabe 1

Welche transformationen sind nötig um den Punkt $p = (2, 4)$ mit dem Winkel $\pi/4$ im Uhrzeigersinn um den Punkt $o = (1, 1)$ zu rotieren?

Schreiben Sie die Transformationen in Matrix/Vektor-Form und mittels homogener Matrizen. Sie können zur Überprüfung Ihrer Ergebnisse den Punkt z.B. mit GNU Octave oder Asymptote entsprechend transformieren.

Übersetzen Sie das Beispiel in Asymptote-Code und prüfen Sie ob die entsprechenden Matrizen (`transform` Instanzen) den Punkt wie gewünscht rotieren (siehe https://asymptote.sourceforge.io/doc/Transforms.html).

Aufgabe 2

Angenommen die Matrix $\mathbf{M}$ ist das Ergebnis der Eulerrotation $\mathbf{R}_X(\alpha)\mathbf{R}_Z(\beta)\mathbf{R}_X(\gamma)$, was sind die Rotationswinkel $\alpha, \beta, \gamma \in [-\pi, \pi]$?

$$\mathbf{M} = \begin{pmatrix} 0 & 0 & 1 \\ -0.5 & -0.86603 & 0 \\ -0.86603 & -0.5 & 0 \end{pmatrix}$$

Hinweis: Es ist nicht nötig die letzte letzte Matrixmultiplikation zuende zu rechnen.

Hinweis: Eulerrotationswinkel sind im Wertebereich $[0, \pi]$ und die verwendeten Winkel sind einfach.

Aufgabe 3

Analysieren Sie die Eulerrotation $\mathbf{R}_Z(\phi)\mathbf{R}_X(\theta)\mathbf{R}_Y(\psi)$, in der Konfiguration $\psi = 0, \theta = \pi/2$, was ist passiert und warum?

Aufgabe 4

Wie würden Sie ein C oder C++ Programm schreiben, das drei Winkel (z.B. über die Kommandozeile angegeben) quantisiert und in einem `uint32_t` speichert. Wie können die Zahlen wiederhergestellt werden?

Was sind die Vor- und Nachteile einer solchen Repräsentation?

Aufgabe 5

Das Asymptote Programm `solar.asy` enthält die Grundlage einer Skizze der Rotation der Erde um die Sonne, sowie des Monds um die Erde. Weiterhin ist ein Weltraumfahrstuhl vorhanden. Implementieren Sie die korrekten Rotationen (unter der Annahme dass der Tag exakt 24 Stunden hat, das Jahr exakt 365 Tage und die Umlaufbahnen zirkulär, nicht elliptisch, sind).

Testen Sie per Hand einzelne Kombinationen auf Plausibilität. Mit der Option `video_test` können Sie viele Bilder in geringer Auflösung generieren. Damit lässt sich überprüfen, ob die Bewegung über Zeit stimmt. Um ein Video aus den Bildern zu rendern können Sie folgende Kommandozeile verwenden:

```
$ cat solar_*png | ffmpeg -f image2pipe -r 30 -i - -vcodec libx264 video.mkv -y
```

Sie können ebnefalls ein Video in hoher Auflösung rendern (`early_out = video_test = false;`), sollten Sie den entsprechenden Enthusiasmus an den Tag legen :)

Hinweis: In diesem Fall werden Png-Bilder erzeugt, der entsprechende Asymptote-Aufruf ist dann `asy -f png trafo.asy`.

Aufgabe 6

Das Asymptote Programm `slerp.asy` zeichnet eine dreidimensionale Kugel wie sie auch in den Vorlesungsfolien zu Slerp zu sehen ist. Das Programm berechnet verschiedene interpolierte Positionen zwischen den beiden Punkten auf der Kugel, wahlweise mit linearer Interpolation (siehe `lerp`, schon implementiert), mittels N-Lerp (siehe `nlerp`) und mittels Slerp (siehe `slerp`), wobei die letzten beiden zu implementieren sind. Vergleichen Sie die Ergebnisse.

Hinweis: In diesem Fall werden Png-Bilder über Asymptotes Rasterisierer erzeugt, der entsprechende Aufruf für Asymptote ist dann `asy -f png -render 8 slerp.asy`.