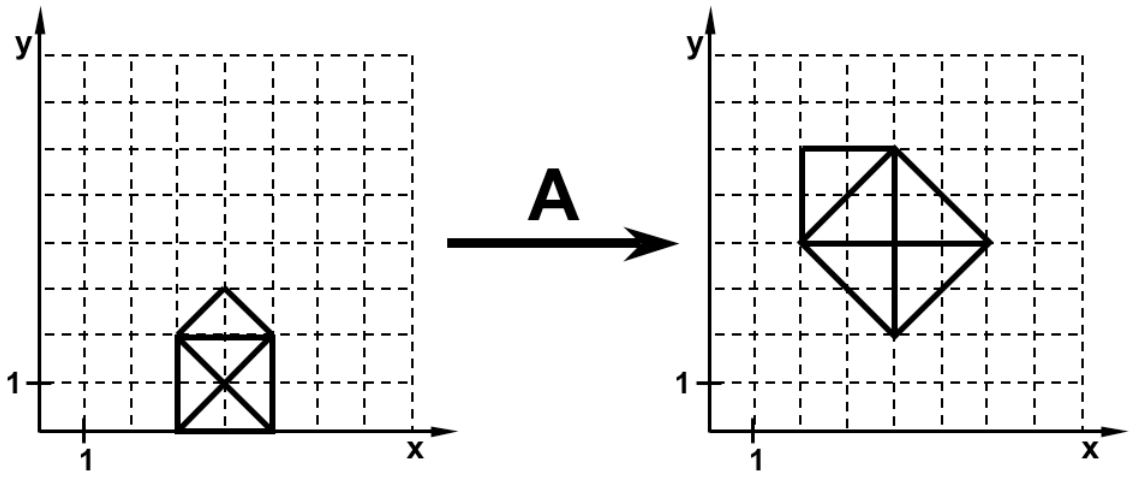
Übungsblatt 5

Willkommen zur fünften Übung zur Vorlesung *Generative Computergrafik*. Dieses Blatt behandelt im Wesentlichen die *Geometrie-Pipeline für Punkte* zur Rasterisierung von Objekten.

Aufgabe 1. Bestimmen Sie die 3×3 Matrix, die der Abbildung **A** entspricht. Überlegen Sie sich hierzu zunächst, aus welchen einfachen Transformationen (Translation, Rotation, Skalierung) sich **A**, in welcher Reihenfolge zusammensetzt.



Aufgabe 2. Es seien die nachfolgenden Quaternionen gegeben.

$$\mathbf{q}_1 = \left[\frac{\sqrt{2}}{2}, \begin{pmatrix} 0 \\ \frac{\sqrt{2}}{2} \\ 0 \end{pmatrix}\right], \mathbf{q}_2 = \left[2, \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}\right], \mathbf{q}_3 = \left[\frac{\sqrt{2}}{2}, \begin{pmatrix} 0 \\ 0 \\ \frac{\sqrt{2}}{2} \end{pmatrix}\right], \mathbf{q}_4 = \left[\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2} \begin{pmatrix} \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{pmatrix}\right]$$

Ermitteln Sie die Rotationsmatrizen zu allen *Einheitsquaternionen* und berechnen Sie weiterhin die folgenden Größen:

1. $\mathbf{q}_1 + \mathbf{q}_3$ und $\mathbf{q}_1 - \mathbf{q}_3$.
2. $\mathbf{q}_1\mathbf{q}_2$, $\mathbf{q}_2\mathbf{q}_1$, $\mathbf{q}_1\mathbf{q}_3$ und $\mathbf{q}_1(\mathbf{q}_2 + \mathbf{q}_3)$.
3. $\bar{\mathbf{q}}_1$, $\bar{\mathbf{q}}_2$, $\bar{\mathbf{q}}_3$, $\bar{\mathbf{q}}_4$.
4. $||\mathbf{q}_1||$, $||\mathbf{q}_2||$ und $||\mathbf{q}_1 \cdot \mathbf{q}_2||$.
5. $\mathbf{q}_1^{-1}$ und $\mathbf{q}_2^{-1}$.

Aufgabe 3. Um welchen Winkel muss der Vektor $\mathbf{v} = (1, 1)^T$ in einem rechtshändigen Koordinantensystem gegen den Uhrzeigersinn gedreht werden, damit er kollinear zu dem Vektor $\mathbf{w} = (2, 1)^T$ ist? *Hinweis:* Verwenden Sie das Vektorprodukt um den richtigen Winkel auszuwählen.

Aufgabe 4. Schreiben Sie einen einfachen *Punkt-Renderer*, also ein Programm zur Darstellung von $3D$-Punktwolken. Nutzen Sie hierfür die Skripte aus dem Archiv `pointViewerTemplate.zip` von der Webseite. Beim Aufruf von

```
python pointViewerTemplate.py objectPoints.obj
```

soll Ihr Programm (nur) die Punkte aus der Datei `objectPoints.obj` einlesen und zunächst einfach mittels orthographischer Parallelprojektion darstellen (siehe Abbildung). Gehen Sie hierzu folgendermaßen vor

1. Berechnen Sie die Boundingbox des eingelesenen Modells.
2. Verschieben Sie den Mittelpunkt der Boundingbox des Modells in den Ursprung und skalieren Sie die Boundingbox (und damit das Modell) anschließend in den Bereich $[-1, 1]^3$.
3. Projizieren Sie das Modell mittels orthographischer Parallelprojektion in den Bereich $[-1, 1]^2$ auf die xy-Ebene.
4. Transformieren Sie den Bild-Bereich $[-1, 1]^2$ in den Viewport-Bereich $[0, 0] \times$ [`Fenterbreite`, `Fenterhöhe`]. Beachten Sie dabei, dass der Nullpunkt des *glfw*-Viewports in der linken unteren Ecke liegt.

Nachdem das Modell angezeigt wurde, soll es weiterhin mit Hilfe der Buttons, bzw. den Tasten

- `x`/`X` in positiver/negativer Richtung um die x-Achse gedreht werden.
- `y`/`Y` in positiver/negativer Richtung um die y-Achse gedreht werden.
- `z`/`Z` in positiver/negativer Richtung um die z-Achse gedreht werden.

Erweitern Sie Ihr Programm schließlich so, dass es mit Hilfe der Taste '`p`' zwischen *orthographischer* und *perspektivischer Projektion* umschaltet.