Aufgabe 1 (19 Punkte), Teil I

- Wie berechnet sich die Länge eines Vektors $v = (x, y, z)$?
- Welche Koordinaten hat der Vertex $(4, 2, 6, 2)$ im R^3?
- Geben Sie einen zu $v = (1,2)$ orthogonalen Vektor an.
- Geben Sie einen dunkelgrünen Farbton in RGB an.
- Zeichnen Sie ein konkaves Polygon.
  - Z.B.:

Aufgabe 1 (19 Punkte), Teil II

- Nennen Sie eine feste Stage der OpenGL Graphics Pipeline.

- Nennen Sie einen Nachteil der lokalen Beleuchtung.

- Nennen Sie einen Vorteil der lokalen Beleuchtung

Aufgabe 1 (19 Punkte), Teil IV

- Was liefert die GLSL Methode reflect und was für Parameter bekommt sie?
  - Hinweis: Zu speziell, würden wir nicht abfragen
- Wie oft wird der Vertex Shader in einem Durchlauf der Graphics Pipeline ausgeführt?
- Wie viele Fragments können pro Pixel minimal und maximal erzeugt werden?
- Geben Sie die Formel zur Winkelberechnung zweier Vektoren an, die das Kreuzprodukt verwendet.

Aufgabe 2 (1+3 Punkte)

- Gegeben seien die beiden Vektoren $v_1 = (3,0,4)$ und $v_2 = (6,8,0)$.
- Berechnen Sie den Kosinus des Schnittwinkels α der beiden Vektoren.
- Geben Sie dabei Ihre Ausgangsformel und den Rechenweg an.
- Lösung:

Aufgabe 3 (1+1+3+5 Punkte)

- Stellen Sie jeweils die 4×4-Matrix auf, die durch die folgenden Transformationen beschrieben wird.
  - $S\left(1, \frac{1}{2}, 2\right)$, eine Skalierung mit den Faktoren $1, \frac{1}{2}$ und 2, jeweils in x-, y- und z-Richtung.
  - $T(-1, -2, -1)$, eine Translation um den Vektor $(-1, -2, -1)$.
  - $R_z\left(\frac{\pi}{2}\right)$, eine Rotation um die z-Achse mit dem Winkel $\frac{\pi}{2}$.
  - $M = S\left(1, \frac{1}{2}, 2\right) \cdot T(-1, -2, -1) \cdot R_z\left(\frac{\pi}{2}\right)$, die Gesamttransformationsmatrix

Aufgabe 3 (1+1+3+5 Punkte)

$S\left(1, \frac{1}{2}, 2\right) =$

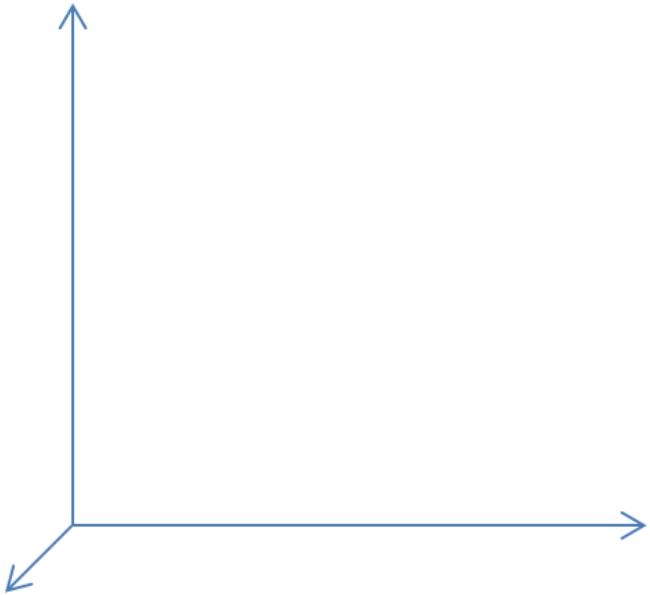
$T(-1, -2, -1) =$

$R_z\left(\frac{\pi}{2}\right) =$

$M = S\left(1, \frac{1}{2}, 2\right) \cdot T(-1, -2, -1) \cdot R_z\left(\frac{\pi}{2}\right)$

Aufgabe 4 (4+2+6 Punkte)

- Zeichnen Sie die folgenden vier Vertices in ein **rechtshändiges** Koordinatensystem ein (x-Achse zeigt nach rechts, y-Achse nach oben) und verbinden Sie sie zu einem Tetraeder. **Hinweis:** Ein Tetraeder ist eine Pyramide mit einer dreieckigen Grundfläche und besteht somit aus exakt **4 Dreiecken**. Jeder Punkt ist mit jedem anderen verbunden.
  - Position: (0,0,-1), Farbe: (1,0,0,1)
  - Position: (1,0,-2), Farbe: (1,1,0,1)
  - Position: (0,1,-2), Farbe: (0,0,1,1)
  - Position: (0,0,-2), Farbe: (0,1,0,1)



Hinweis: Zeichnung nicht maßstabsgetreu, muss in der richtigen Klausur auch nicht

Aufgabe 5a (~~3~~ 2 Punkte)

- Erweitern Sie Abbildung 1 um diejenigen Vektoren, die zur Berechnung des diffusen Anteils des Phong Beleuchtungsmodells benötigt werden. Markieren Sie außerdem den Winkel, der letztendlich für die Stärke der diffusen Beleuchtung verantwortlich ist.

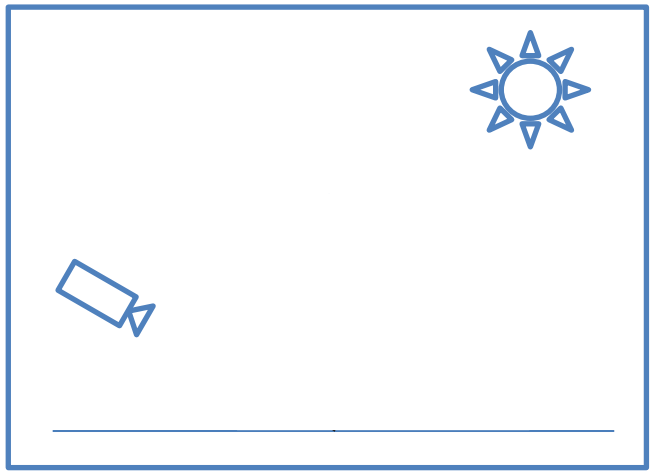


Abbildung 1

Aufgabe 5b (4 3 Punkte)

- Erweitern Sie Abbildung 2 um diejenigen Vektoren, die zur Berechnung des spekularen Anteils des Phong Beleuchtungsmodells benötigt werden. Markieren Sie außerdem den Winkel, der letztendlich für die Stärke der spekularen Beleuchtung verantwortlich ist.

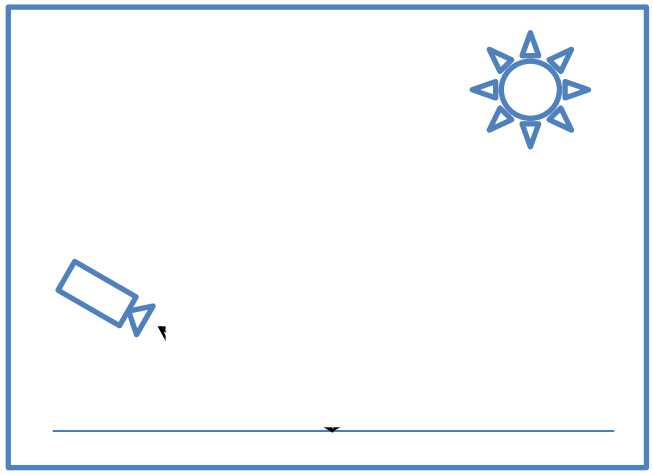


Abbildung 2

Aufgabe 5c (~~5~~ 7 Punkte)

- Erweitern Sie die GLSL Methode vec4 Enlight(vec3 position, vec3 normal), die sowohl den diffusen als auch den spekularen Beleuchtungsanteil *(des Phong Modells)* berechnet und deren Summe zurückgibt. Die Entfernung und Art der Lichtquelle sollen dabei unberücksichtigt bleiben

```
uniform vec3 lightPosition;
uniform vec3 eyePosition;
uniform vec3 I_d, I_s, c_d, c_s, k_d, k_s;
uniform float es;

vec3 Enlight(vec3 position, vec3 normal){ // normal sei normiert









}
```

Aufgabe 7 (3+3+3 Punkte)

- Wir sind uns in einem Programm nicht sicher, ob die Normalen einer Geometrie korrekt berechnet wurden. Um dies zu prüfen, würden wir gerne die Fragments entsprechend ihrer Normale einfärben. Implementieren Sie dazu den Vertex- und den Fragmentshader. **Achtung:** Die Einträge des Normalenvektors können negativ sein. Sorgen Sie für eine entsprechende Normierung.

Aufgabe 7 (3+3+3 Punkte)

VS

```
uniform mat4 viewProjection, model;

uniform mat3 model_Normals;
in vec3 positionMC;
in vec3 normalMC;


void main(){ // Klar, was im VS geschehen muss?



}
```

FS

```
void main(){


}
```

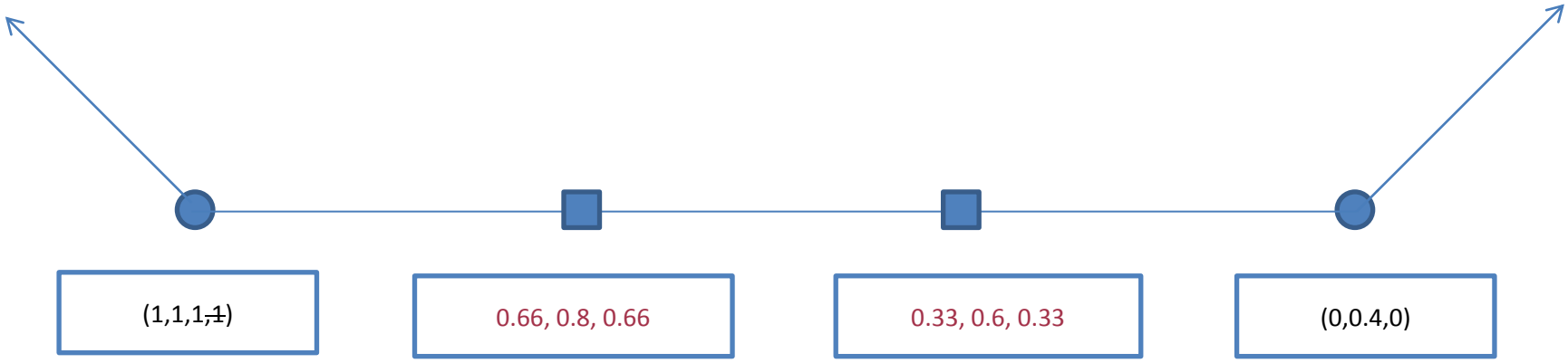
- Die Farben entsprechen der Richtung der Normalen. Sehen wir bspw. ein blaues Fragment, heißt das, dass die Normale genau in z-Richtung zeigt, also (0,0,1). Ein schwarzes Fragment würde aufgrund der Normierung einer Normale von (-1,-1,-1) entsprechen.

2

Alternative Aufgaben (Keine vollständige Klausur)

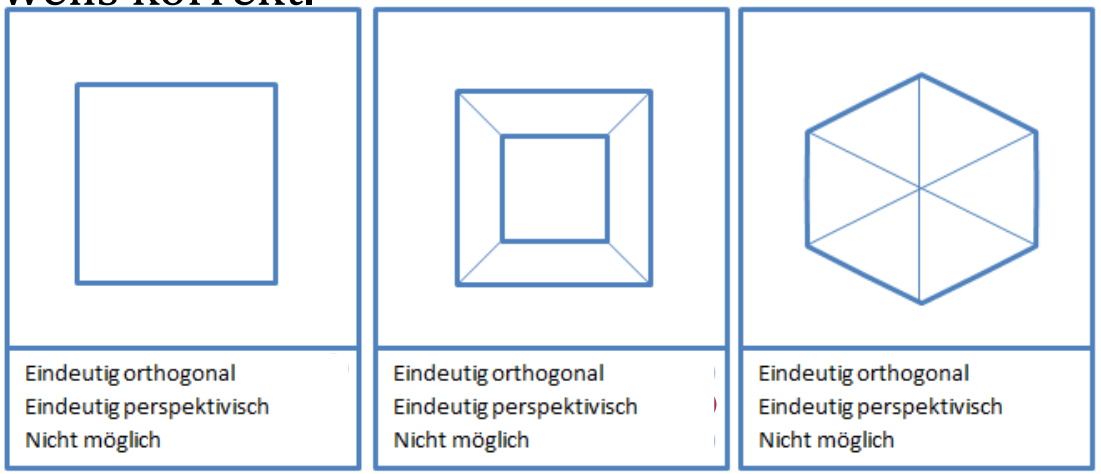
Alternativaufgabe 1 (8 Punkte)

- In der Zeichnung unten sind zwei Vertices (●) jeweils mit einer Normalen und einer Farbe angegeben.
- Der Rasterizer erzeugt die beiden Fragments (■).
- Zeichnen Sie an diesen Stellen jeweils die interpolierte Normale ein und berechnen Sie die interpolierte Farbe.

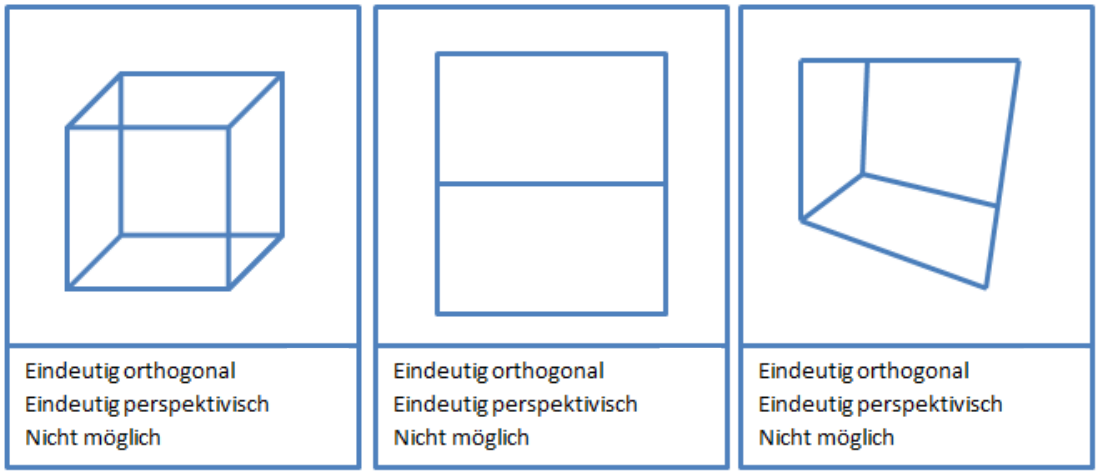


Alternativaufgabe 1 (6 Punkte)

- Gegeben sei ein perfekter Würfel. Ordnen sie seinen Darstellungen als Drahtgitter unten jeweils die Art der Projektion zu. **Hinweis:** Genau eine Antwort ist jeweils korrekt.



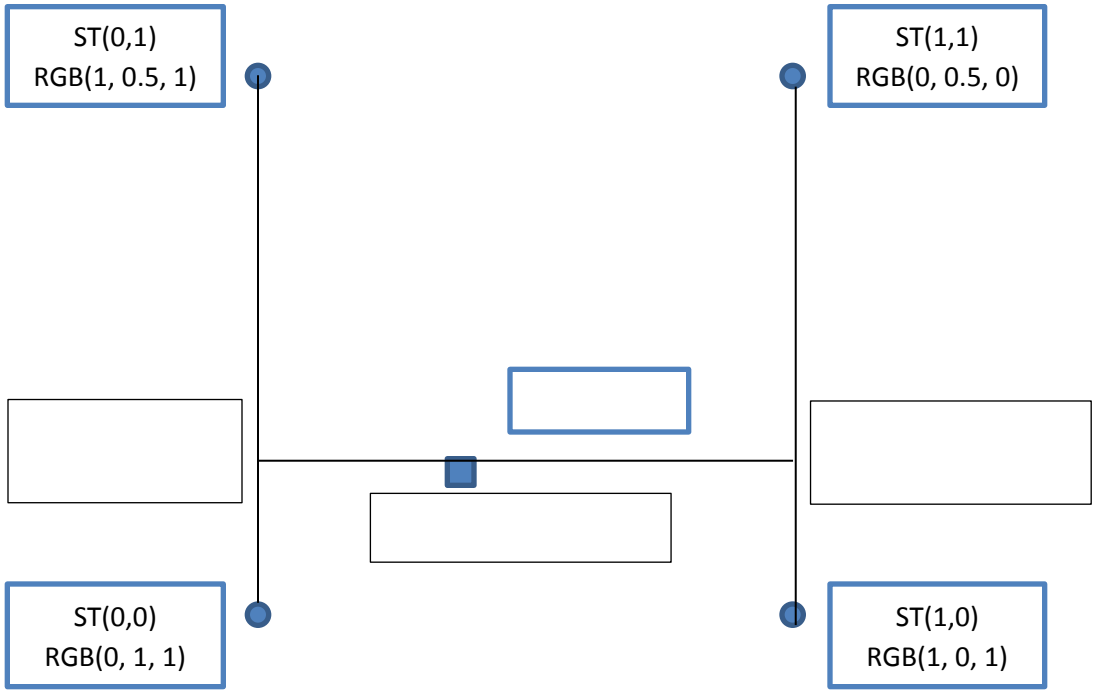
Eindeutig orthogonal Eindeutig perspektivisch Nicht möglich	Eindeutig orthogonal Eindeutig perspektivisch Nicht möglich	Eindeutig orthogonal Eindeutig perspektivisch Nicht möglich



Eindeutig orthogonal Eindeutig perspektivisch Nicht möglich	Eindeutig orthogonal Eindeutig perspektivisch Nicht möglich	Eindeutig orthogonal Eindeutig perspektivisch Nicht möglich

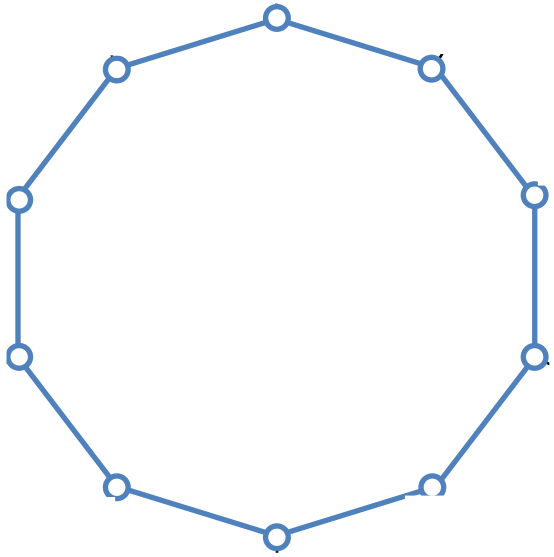
Alternativaufgabe 3 (10 Punkte)

- Führen Sie bilineares Filtern aus. ST gibt dabei die Koordinaten der Texel an und RGB ihre Farbe. Gesucht ist die Farbe des Quadrats. Zeichnen Sie die Linien ein, auf denen Sie linear interpolieren, sowie die interpolierten Werte, um den Farbwert des Quadrats zu erreichen. *Hinweis: Interpolieren Sie in t-Richtung zuerst*



Zusatzaufgabe 6 (2 Punkte)

- Wir möchten einen Kreis mittels Rastergrafik darstellen. Dafür nähern wir die Geometrie durch 10 Vertices an.
- Zeichnen Sie an jedem Vertex seine zugehörige Normale ein, die der angestrebten Geometrie am ehesten entspricht.



Zusatzaufgabe 7 (3 Punkte)

- Zeichnen Sie folgende Begriffe in die OpenGL Graphics Pipeline ein.
  - Rasterizer
  - Fragment Shader
  - Vertex Shader
  - Primitve Assembly
  - Per Fragment Operations
  - Primitive Processing