Aufgabe 1

a)

Color Matching Funktionen

Berechnungsvorschrift eines Tristimuluswerts für Color Matching Funktion und Spektrum :

b)

Aus der Berechnungsweise der Tristimuluswerte folgt, dass mehrere unterschiedliche Spektren denselben Wert erzeugen können. Daraus folgt, dass eine Rückkonvertierung nicht eindeutig möglich ist. Diese unterschiedlichen Spektren nennt man Metamerismen.

c)

Der Mensch nimmt Luminanz bei hohen Frequenzen stärker wahr als Chrominanz.

Wenn man also die beiden Teile unterschiedlich behandelt und die Luminanz

weniger stark komprimiert als die Chrominanz.

d)

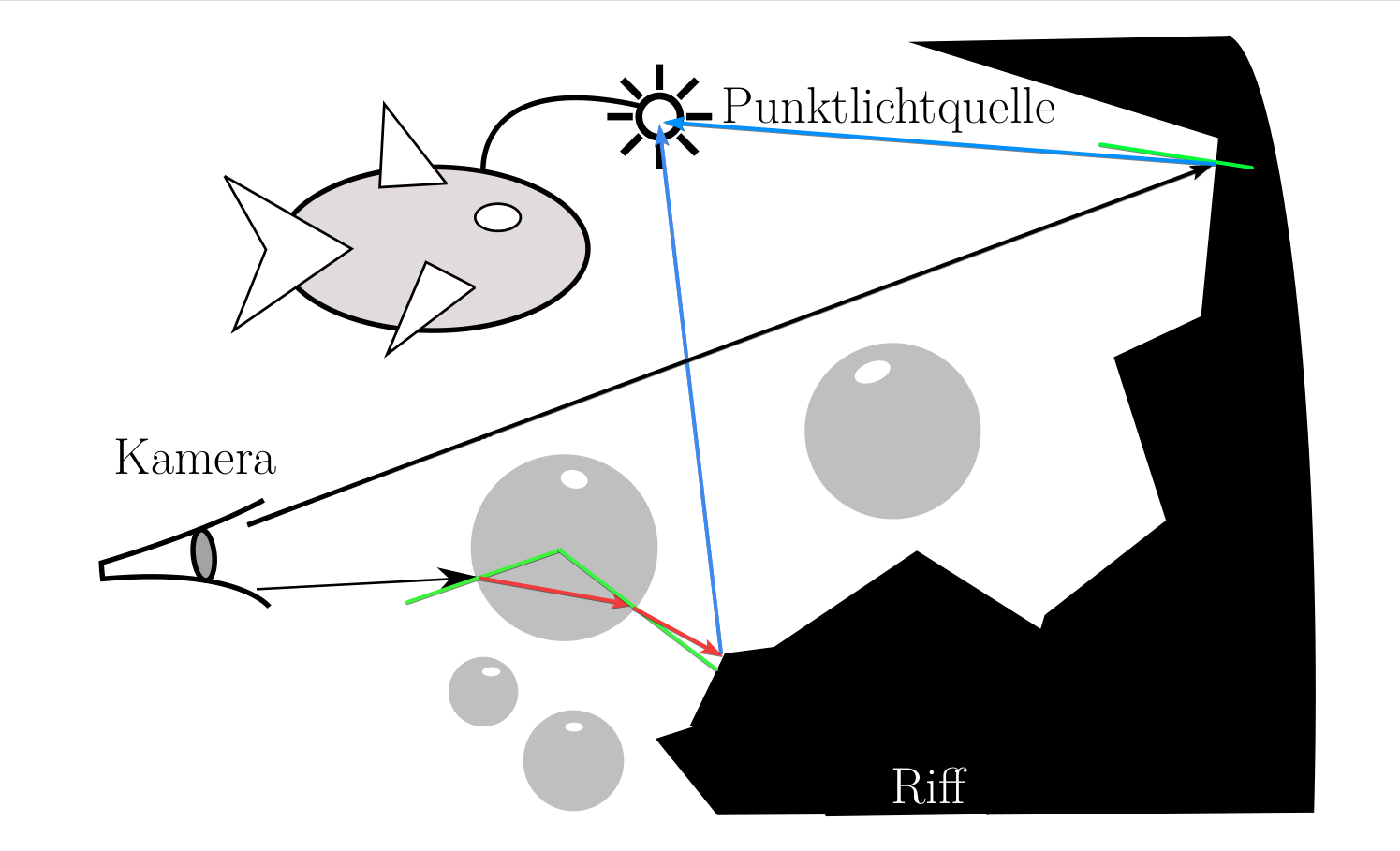
Der wahrgenommene Kontrast nimmt nach dem Weber-Fechner-Gesetz ab.

Aufgabe 2

a)

Frage: Sind Schattenstrahlen an reflektierenden oder transmittierenden Flächen

sinnvoll?



Normaler Strahl

Normale

Transmission

Schatten

b)

4: 0 für die Primärstrahlen + maximal 4 weitere, wenn man durch beide Blasen muss

2

c)

Terminiert, weil es keine Spiegel gibt.

Abbruchkriterien:

* Nach bestimmter Rekursionstiefe
* Wenn der Anteil des Strahls an der Farbe des Originalpixels einen bestimmten Wert unterschreitet

Aufgabe 3

a)

AABB (einfachere Berechnung) und OBB (Genauigkeit, ist kleiner)

b)

Surface Area Heuristic Formel C=..., Abbruch wenn C < …

Skript Seite 108

c)

i.

BSP: Schnittebenen sind nicht achsenparallel sondern beliebig orientiert, kein Gitter und es lässt sich keine AABB erkennen

ii.

1. a
2. b (unterer Teil von b wird verworfen, da Schnittpunkt nach dem Schnittpunkt von a)
3. d
4. f (unterer Teil von f wird verworfen, siehe 2)
5. A
6. (backtrack) g (oberer Teil wird verworfen, da Schnittpunkt vor Schnittpunkt mit d)
7. C, E, H
8. (backtrack) c (oberer Teil wird verworfen, siehe 6)
9. i
10. L, H
11. H, K

Aufgabe 4

a)

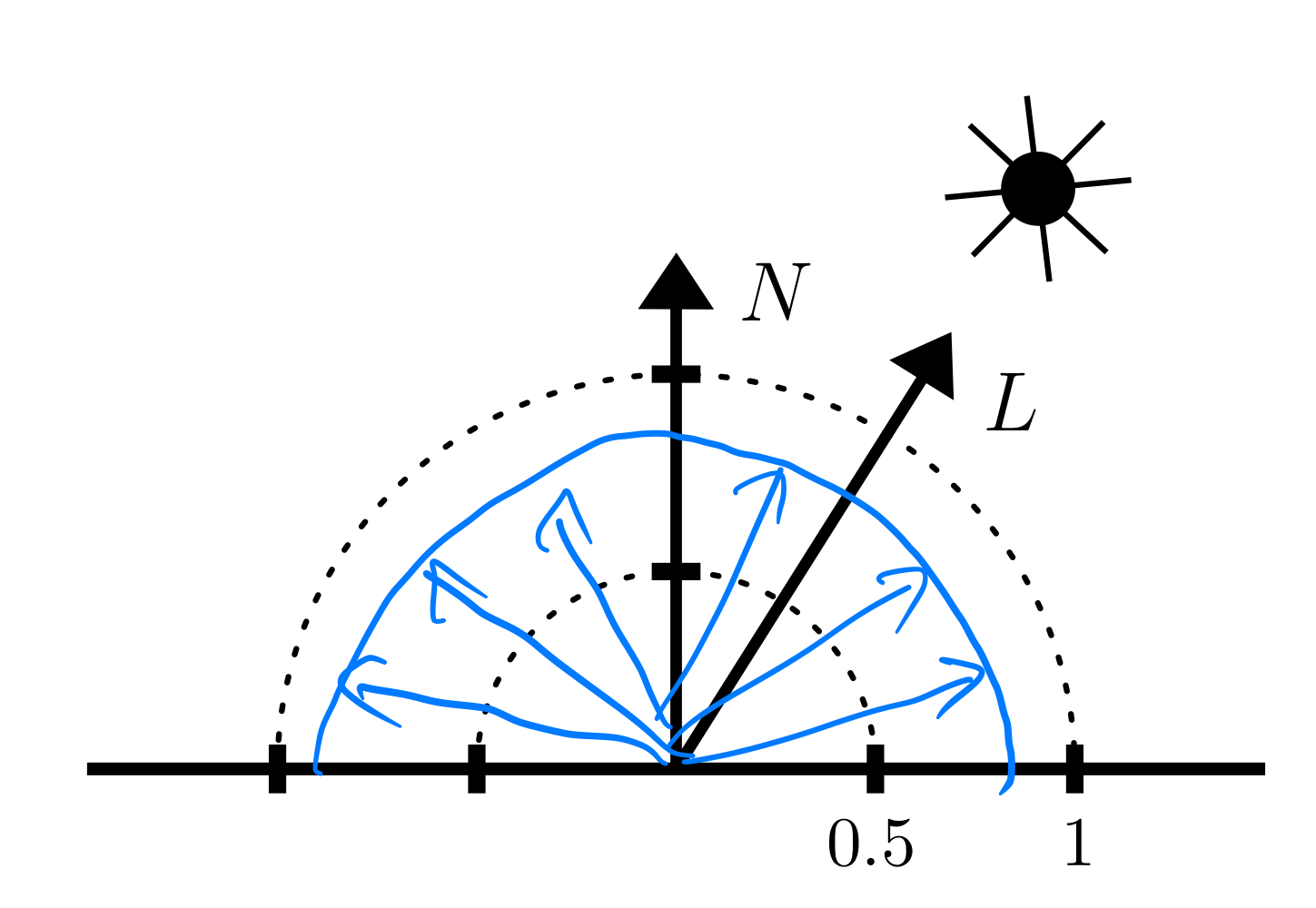
i)

Ambient: 0

Diffus: 1

Spekular: 0

Ergebnis: Kreis mit Radius ungefähr cos(30°)=0.87



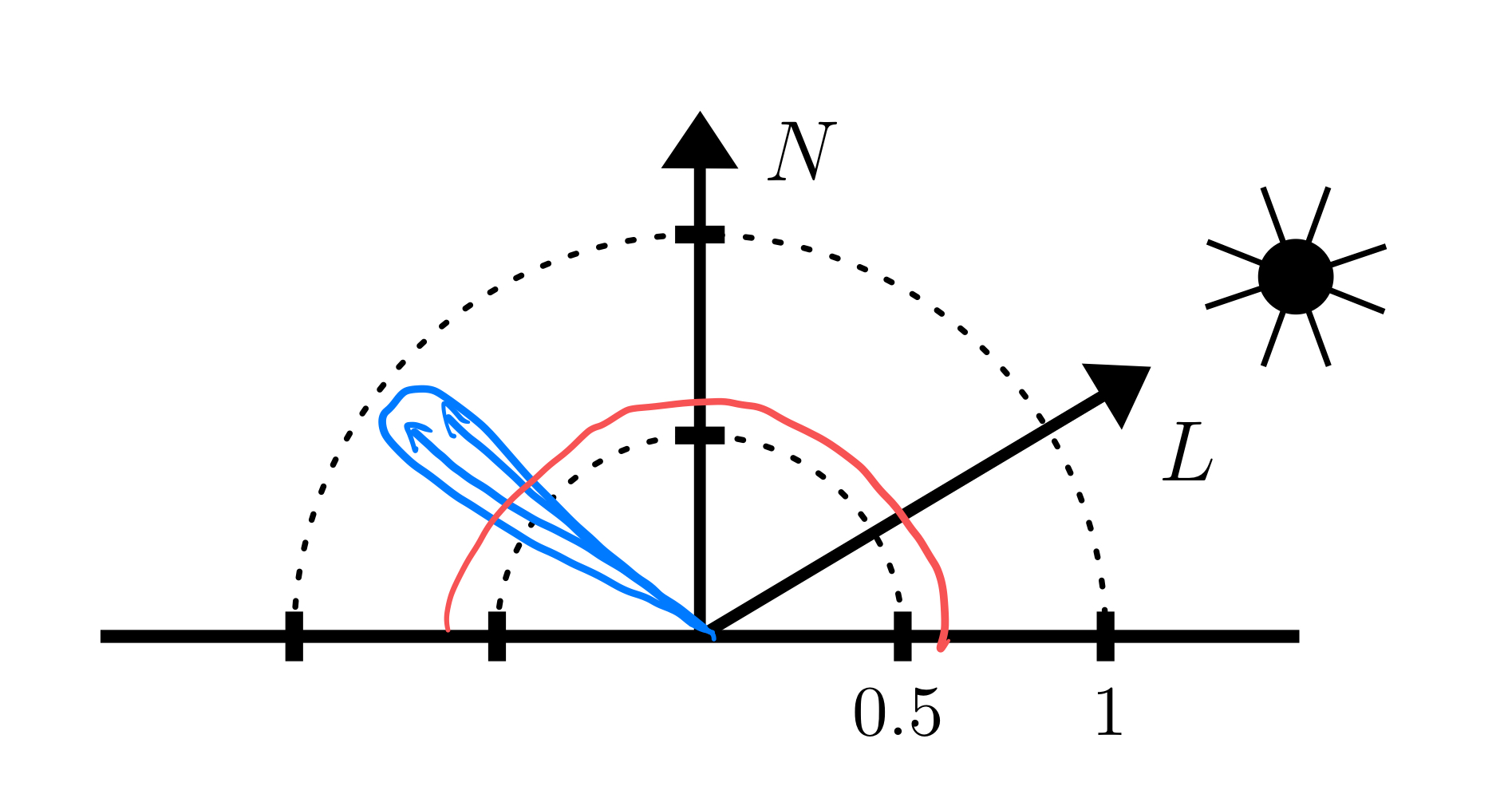
ii)

Ambient: 0.6

Diffus: 0

Spekular: 0.4 mit gigantischem Phong Exponent

Ergebnis: Kreis mit Radius 0.6 + starker Peak zu Radius 1 beim reflektierten Licht



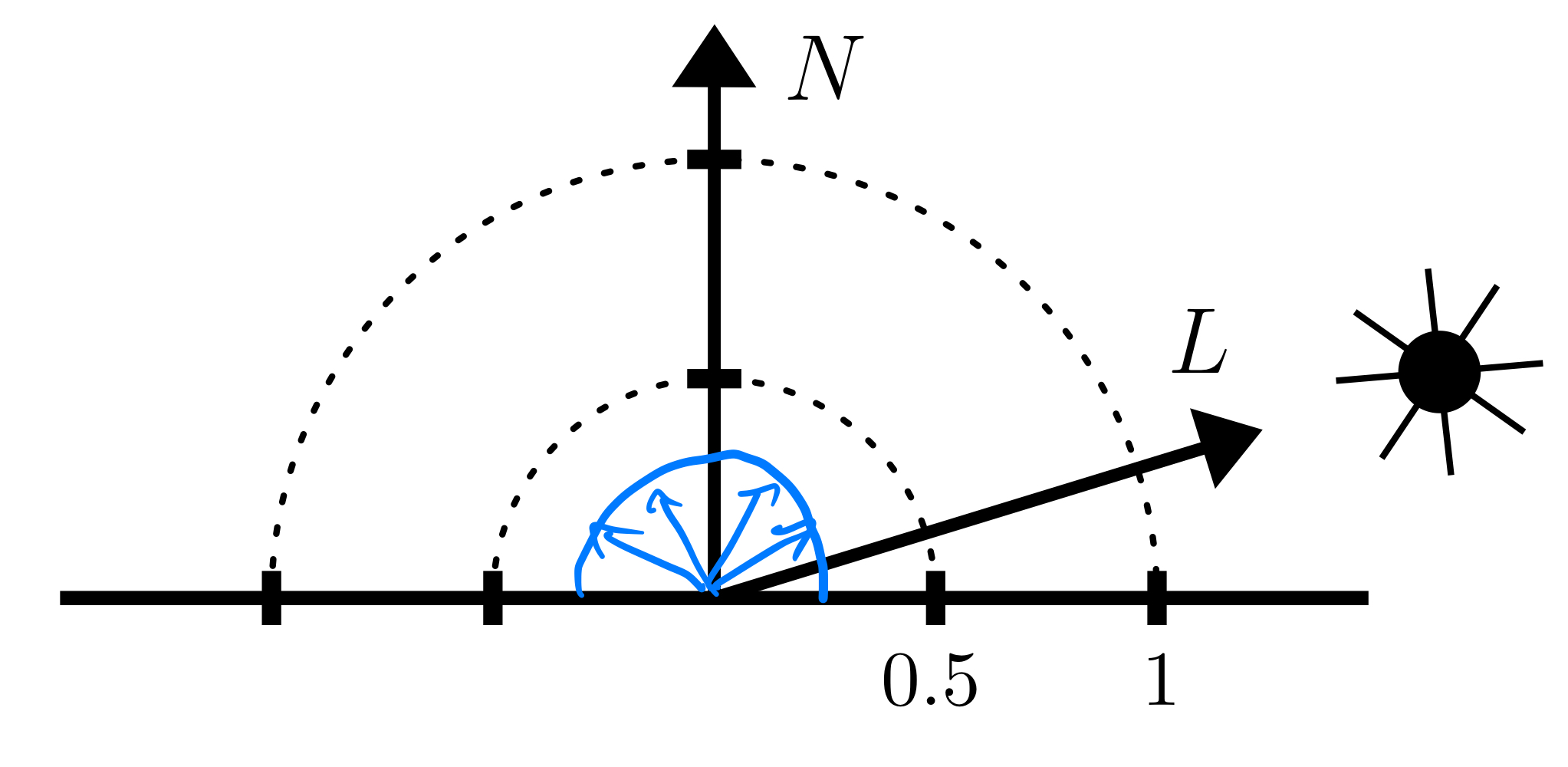
iii)

Ambient: 0

Diffus: 1

Spekular: 0

Ergebnis: Diffus mit sehr flachem Winkel (ca 75°), Radius ungefähr cos(75°)=0.26



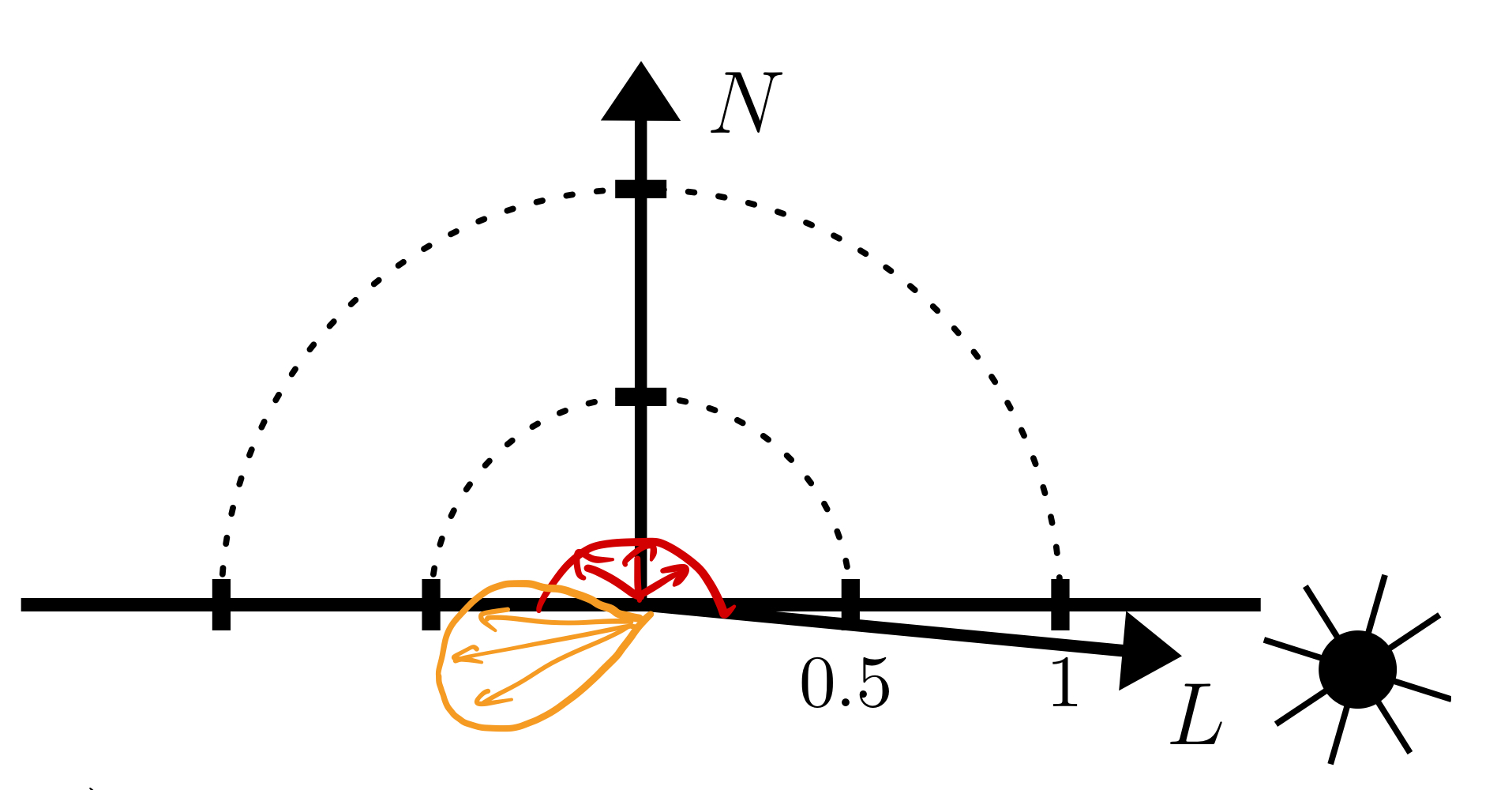
iv)

Ambient: 0.2

Diffus: 0 (negatives Skalarprodukt)

Spekular: 0.5

Ergebnis: Kreis mit Radius 0.2 + relativ flacher Peak zur 0,7 in der Nähe des reflektierten Lichtvektors



b)

i) Phong-Shading

ii) Die interpolierte Normale muss noch normalisiert werden.

iii) mehrere Normalen pro Vertex, berücksichtige nur Normalen deren Winkel zu Dreiecksnormalen nicht zu groß ist

iv) Normalen aus benachbarten Dreiecken addieren und normieren

Aufgabe 5

a)

sx 0 0 0

0 sy 0 0

0 0 sz 0

0 0 0 1

b)

1 0 0 tx

0 1 0 ty

0 0 1 tz

0 0 0 1

c)

gx 0 0 cx

0 gy 0 cy

0 0 gz cz

0 0 0 1

d)

i)

o’: w=1, Matrixmultiplikation, dehomogenisieren

d’: w=0, Matrixmultiplikation, w entfernen weil d ein richtungsvektor und kein ortsvektor ist. Deswegen keine translation

o’ = M^-1 \* o

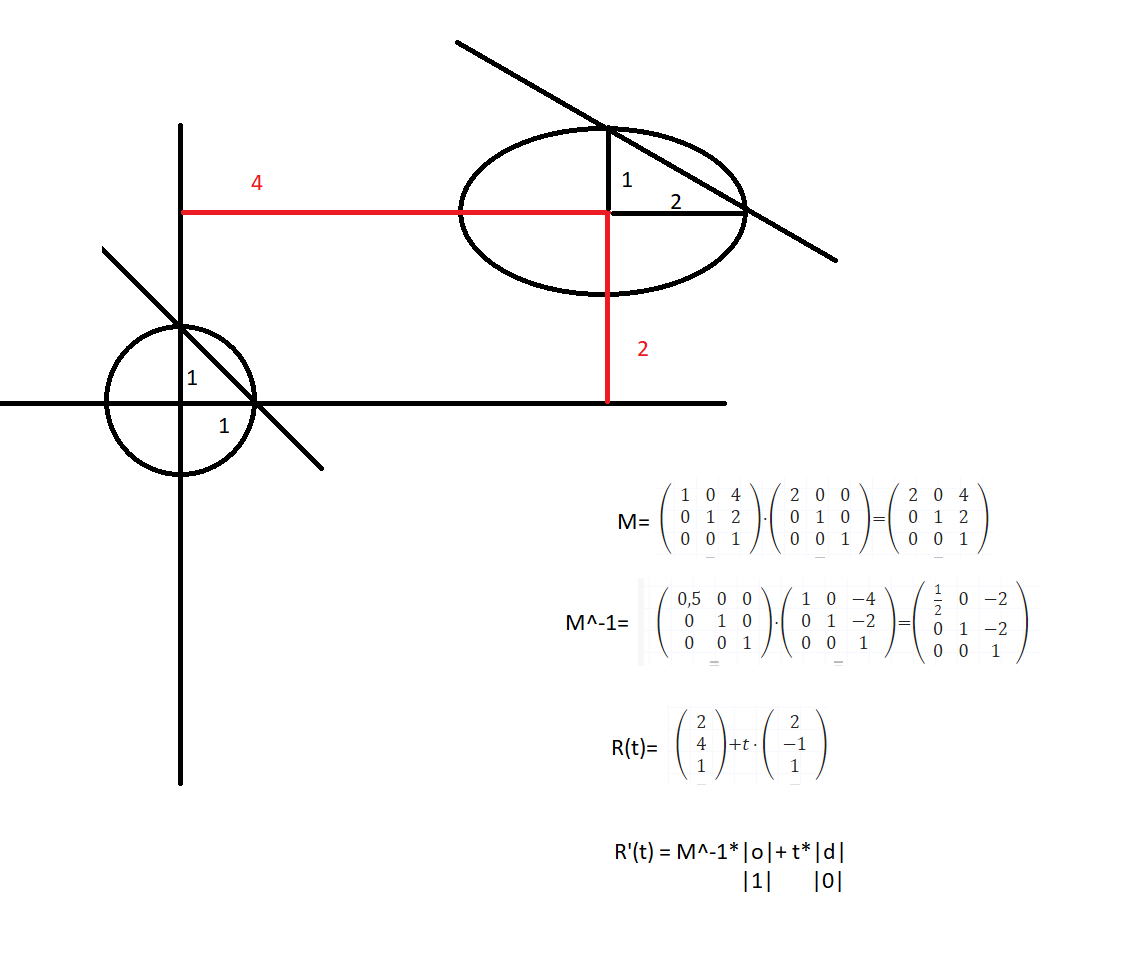
M transformiert die Einheitskugel in den Elipsoid, um von der Welt in das lokale Koordinatensystem zu kommen, muss man M invertieren.

ii)

Wenn d’ nicht normalisiert wurde (muss man d’ nach i) normalisieren?), gilt t’=t und für den Schnittpunkt S=o+t’\*d

nicht ganz, siehe Übungsblatt 3. Der Schnittpunkt mit Einheitskugel soll ins andere Koordinatensystem transformiert werden. Danach gilt t’ nicht mehr. t = length(o - transformierter Schnittpunkt).

Wenn weder d noch d’ normalisiert sind, würde eigentlich t = t’ \* length(t’) / length(t) gelten, oder? | Dann transformiert man halt den Schnittpunkt zurück und setzt t = || o - schnittpunkt ||



mit beispiel

Aufgabe 6

a)

x1 x2 x3

y1 y2 y3

z1 z2 z3

(P1

P2

P3)

b)

l1 = 1 - l2 - l3 ?

c)

Ist das nicht einfach “M(1 - l2’ - l3’, l2’, l3’)”

d)

Könnte bilineare Interpolation sein, macht aber irgendwie wegen “Differenz” in der Aufgabenstellung keinen Sinn

Größe des Footprints für Mip-Mapping vielleicht?

Vielleicht wurden hier Surface Area Tables gemeint?

Displacement Mapping. Aus den Differenzen werden die Normalen bestimmt!

e)

Vorfilterung entfernt hohe Frequenzen, wodurch das (Nyquist-Shannon-)Abtasttheorem erfüllt wird.

Aufgabe 7

a)

Front Buffer, der vom Bildschirm gelesen wird

Back Buffer, in den die Graka das neue Bild schreibt, wenn fertig swap mit Front Buffer

Erlaubt asynchrones Berechnen des nächsten Bilds, ohne Screen-Tearing

b)

Clip Coordinaten, da nach der Normalisierung

* Ambiguitäten bei Linien auftreten können, die mittels Tiefenwerten behoben werden müssten (teuer)
* es kann auch passieren, dass w = −z = 0 ist → Division durch 0? (k.A. wie w=0 passieren soll) siehe Foliensatz 6 S.35

c)

VS: eine Position ausgeben

GS: min. ein Primitiv ausgeben

FS: eine Farbe ausgeben (genau genommen muss nur ein Fragment ausgegeben werden)

d)

Siehe Foliensatz 7 S.174, z.B.

* Stencil Test
* Depth Test
* Alpha Test
* Blending
* Scissor Test

Aufgabe 8

a)

R = Rs \* Rs + (1 - Rs) \* Rd

G = Gs \* Gs + (1 - Gs) \* Gd

B = Bs \* Bs + (1 - Bs) \* Bd

A = As \* As + (1 - As) \* Ad

b)

Für R, andere Farben sind analog:

R1 op R2 = R1 \* R1 + (1 - R1) \* R2

R2 op R1 = R2 \* R2 + (1 - R2) \* R1

offensichtlich gilt R1 op R2 != R2 op R1

c)

Zeichenreihenfolge macht einen Unterschied

d)

* glBlendFunc( GL\_DST\_COLOR, GL\_ZERO ), Verwendung: multiplikative Blending
* glBlendFunc( GL\_ONE, GL\_ONE ); Verwendung: additive Blending, zB wenn mehrere Lichtquellen addiert werden sollen

Verwendung?

z.B. beim Blenden von einer Lichtquelle wie Feuer mit dem Hintergrund

Aufgabe 9

a)

// Nicht failen, da sonst Fragment nicht gezeichnet wird

glStencilFunc(GL\_ALWAYS, 1, 1);

// Stencil Test kann nicht failen (Wert also egal)x

// Depth Test fail egal

// Depth Test success => Ersetze mit 1

glStencilOp(GL\_KEEP, GL\_KEEP, GL\_REPLACE);

b)

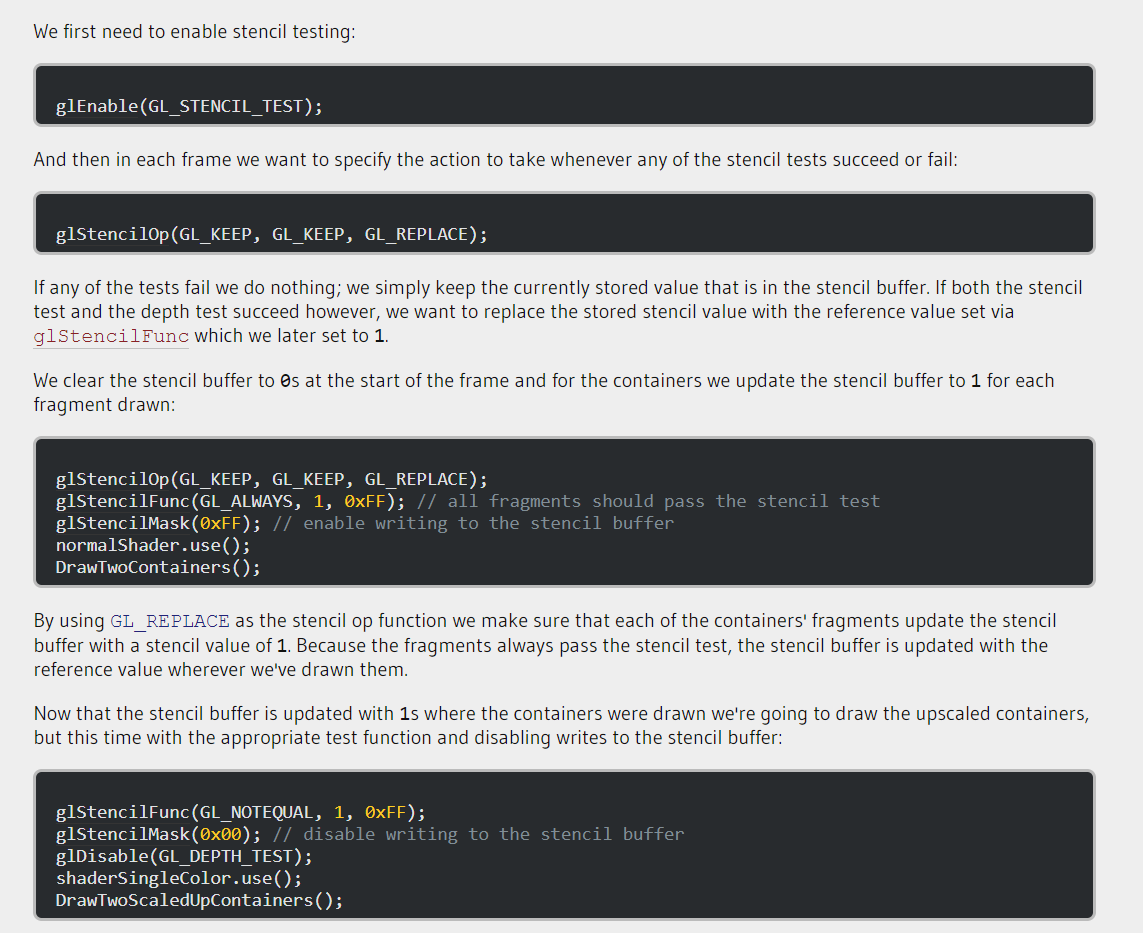
// Faile für Werte gleich 0, Maske muss trotzdem bleiben

glStencilFunc(GL\_EQUAL, 0, 1);

// Hier soll nichts geupdatet werden

glStencilOp(GL\_KEEP, GL\_KEEP, GL\_KEEP);

// Kann das jemand genauer ausführen?

a + b aus learnopengl.com:  


Aufgabe 10

a)

// <= wichtig, da wir Pixel übermalen wollen

glDepthFunc(GL\_LEQUAL);

glBlendFunc(GL\_DST\_ALPHA, GL\_ONE\_MINUS\_DST\_ALPHA);

falls der fresnel faktor im 2. durchlauf gesetzt wird, kann man hier auch folgendes verwenden, oder?

glBlendFunc(GL\_SRC\_ALPHA, GL\_ONE\_MINUS\_SRC\_ALPHA);

b)

Vertex Shader:

out vec3 world\_position;

out vec3 world\_normal;

void main() {

gl\_Position = matrixMVP \* in\_position;

world\_position = (matrixMW \* in\_position).xyz;

// Warum auch immer matrixNrml 4x4 ist

world\_normal = normalize((matrixNrml \* vec4(in\_normal)).xyz);

}

Pixel Shader 1:

in vec3 world\_position;

in vec3 world\_normal;

void main() {

vec3 camToPosVector = normalize(world\_position - camPos);

vec3 lightVector = normalize(lightPos - world\_position);

vec3 reflected = reflect(camToPosVector, world\_normal);

float fresnel = fresnel(world\_normal, reflected);

out\_color = vec4(computeShading(world\_position, world\_normal, lightVector, reflected), fresnel);

}

Pixel Shader 2:

in vec3 world\_position;

in vec3 world\_normal;

void main() {

vec3 lightToPosVector = normalize(world\_position - lightPos);

vec3 reflected = reflect(lightVector, world\_normal);

out\_color = texture(samCube, reflected);

}

Alternative Shader 2:

void main() {

vec3 V = normalize(world\_position - camPos);

vec3 R = reflect(V, world\_normal);

float fresnel = fresnel(world\_normal, R);

out\_color = vec4(0.0);

if (fresnel > 0) {

out\_color = vec4(texture(samCube, R), fresnel);

}

}

Aufgabe 11

a)

* Keine Übertragung von Daten
* Auflösungsunabhängig
* Kompakte Beschreibung

b)

Vielleicht:

* Varianz haben
* keine Periodizität (zumindest nicht sichtbar)
* deterministisch (also nur pseudo-, nicht zufällig)
* reproduzierbar: 𝑛(𝐱)liefert für jede Auswertung bei gleichem 𝐱dasselbe Resultat
* begrenzter Wertebereich (Abb. auf Farben etc.)
* definierte Frequenzverteilung, bandlimitiert (Aliasing reduzieren)
* Stetigkeit, räumliche Korrelation: 𝑛(𝐱)≈𝑛(𝐱+𝛜)(hängt mit Bandbegrenzung zusammen)

(08-40)

c)

turbulence\_f(x) = sum\_i = 0^(n-1) (1/f^i) noise(f^i \* x)

d)

Der genaue Wert kommt auf die Frequenz an, für f=2 z.B. eine weitere Iteration.

Im Allgemeinen für 1 < f aber O(1) zusätzliche Arbeit.

Aufgabe 12

a)

C1 stetig ⇔ b1 - b0 = a3 - a2

b0 = a3

b1 = 2\*a3-a2

b)

Punkt liegt bei (21,6) wenn ich mich nicht verrechnet habe

Bitte Rechenweg angeben

