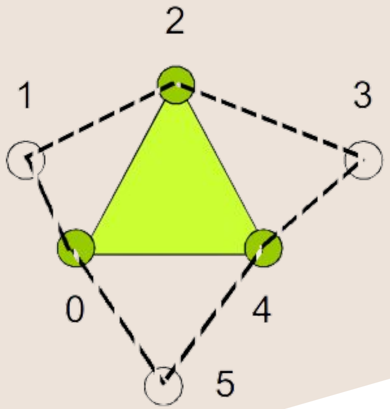
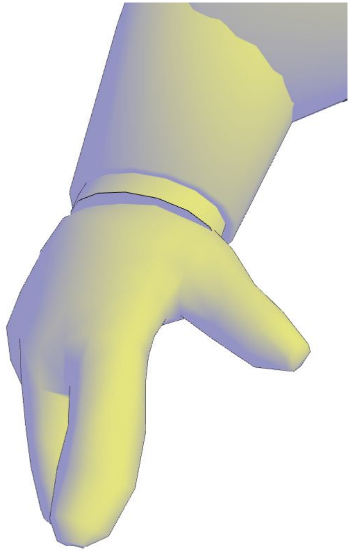
Non-photorealistic rendering

Diogo Vieira	pg50518
Laura Rodrigues	pg50542
Mariana Marques	pg50633

Outlines – creases

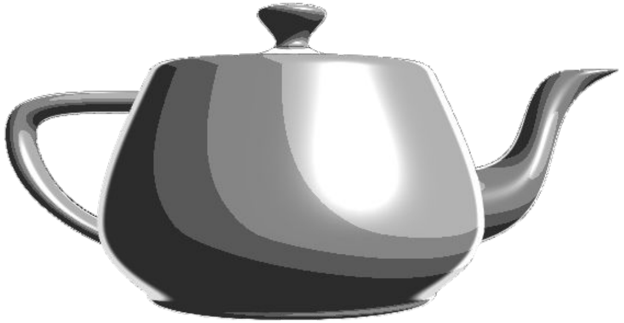
```
    vec3 c = normalize(camPos - (ps[0]+ps[2]+ps[4])/3) ;
    ...
    vec3 N042 = cross(ps[4] - ps[0], ps[2] - ps[0]);
    ...
    float dotView = dot(N042, c);
    if (dotView < 0.0){
        dotView = dot(N021, c2);
        if (dotView >= 0)
            EmitLine(0,2);
        dotView = dot(N243,c4);
        if (dotView >= 0)
            EmitLine(2,4);
        dotView = dot(N405,c6);
        if (dotView >= 0)
            EmitLine(4,0);
}
```





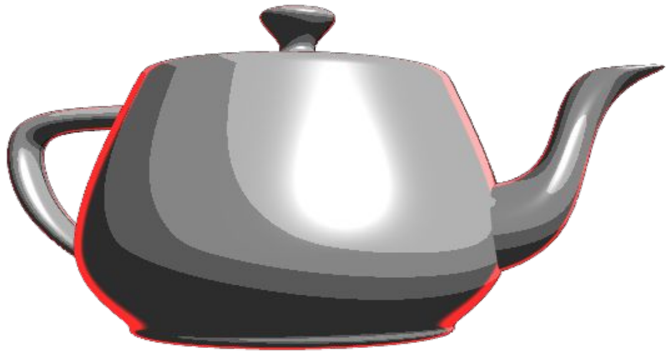
Toon Shading

```
float division = 1.0 / float(num_divisions);
i = ceil((i +0.00001) * num_divisions) * division;
```



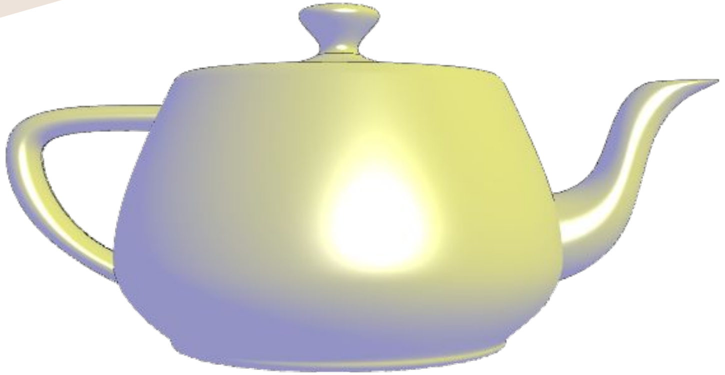
Rim Lighting

```
float rimLightIntensity = dot(ee, n);
rimLightIntensity = 1.0 - rimLightIntensity;
rimLightIntensity = max(0.0, rimLightIntensity);

rimLightIntensity = smoothstep(rimLight,rimLight + 0.1,rimLightIntensity);
vec4 rimLight   = rimLightIntensity * vec4(1.0, 0, 0, 1.0);
```

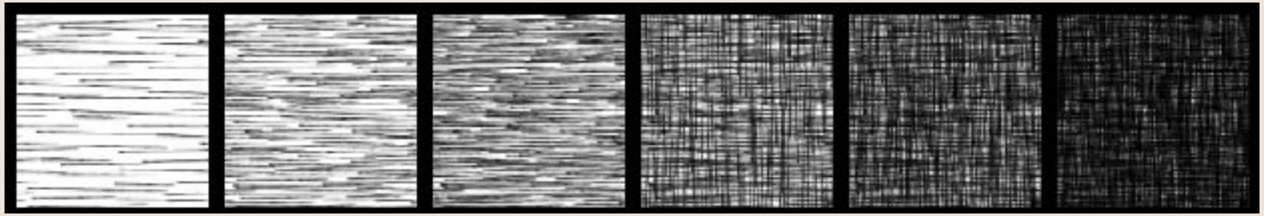


Gooch Shading

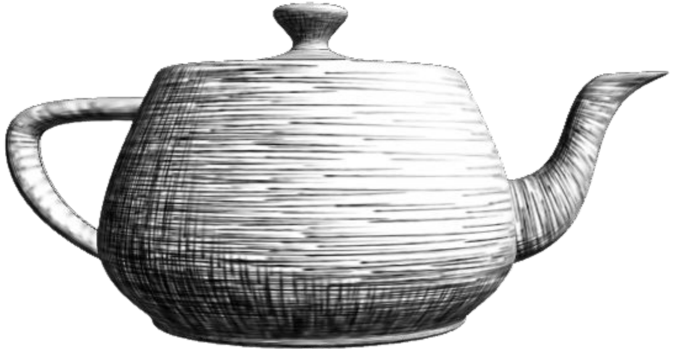
```
vec3 finalCool = coolColor + alpha * vec3(1);
vec3 finalWarm = warmColor + beta * vec3(1);

float lerp = (1.0 + i) / 2.0;
finalCool = (1 - lerp) * finalCool;
finalWarm = lerp * finalWarm;


...


outColor = vec4(finalCool + finalWarm, 1.0) + spec;
```



Hatching

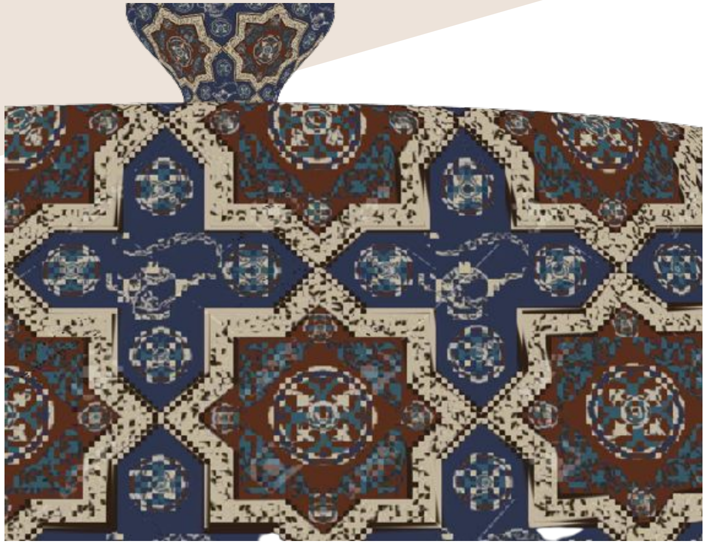


```
float step = 1. / 6.;
if( i <= step )
    c = mix( tex5, tex4, 6. * i );
if( i > step && i <= 2. * step )
    c = mix( tex4, tex3, 6. * ( i - step ) );
if( i > 2. * step && i <= 3. * step )
    c = mix( tex3, tex2, 6. * ( i - 2. * step ) );
if( i > 3. * step && i <= 4. * step )
    c = mix( tex2, tex1, 6. * ( i - 3. * step ) );
if( i > 4. * step && i <= 5. * step )
    c = mix( tex1, tex0, 6. * ( i - 4. * step ) );
if( i > 5. * step )
    c = mix( tex0, vec4( 1. ), 6. * ( i - 5. * step ) );
```



Pixelation

```
int pixelSize = 9;
float x = int(gl_FragCoord.x) % pixelSize;
float y = int(gl_FragCoord.y) % pixelSize;
x = floor(pixelSize / 2.0) - x;
y = floor(pixelSize / 2.0) - y;
x = gl_FragCoord.x + x;
y = gl_FragCoord.y + y;
vec2 uv = vec2(x, y) / texSize;
color = texture(tex, texCoord + uv);
```

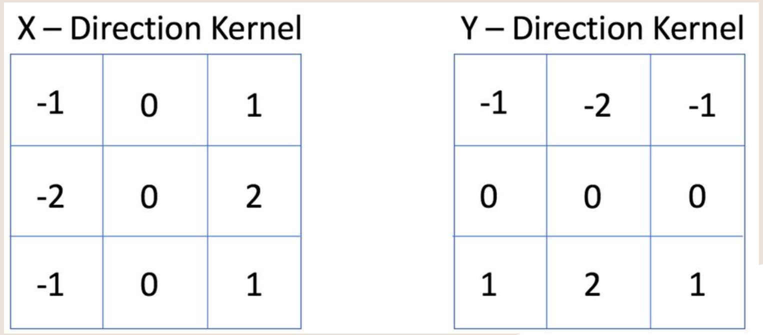


Sobel – Outlines

```
depth = length(vec3(pos)) / (far-near);
```

```
vec4 n[9];
make_kernel( n,tex, texCoordV);
vec4 sobel_edge_h = n[2] + (2.0*n[5]) + n[8] - (n[0] + (2.0*n[3]) + n[6]);
vec4 sobel_edge_v = n[0] + (2.0*n[1]) + n[2] - (n[6] + (2.0*n[7]) + n[8]);
vec4 sobel = sqrt((sobel_edge_h * sobel_edge_h) + (sobel_edge_v * sobel_edge_v));
color = vec4(1- sobel.rgb,1.0 )
```

```
<texture UNIT="1" name="sobel" >
```



X – Direction Kernel

-1	0	1
-2	0	2
-1	0	1

Y – Direction Kernel

-1	-2	-1
0	0	0
1	2	1

Conclusões e Trabalho Futuro