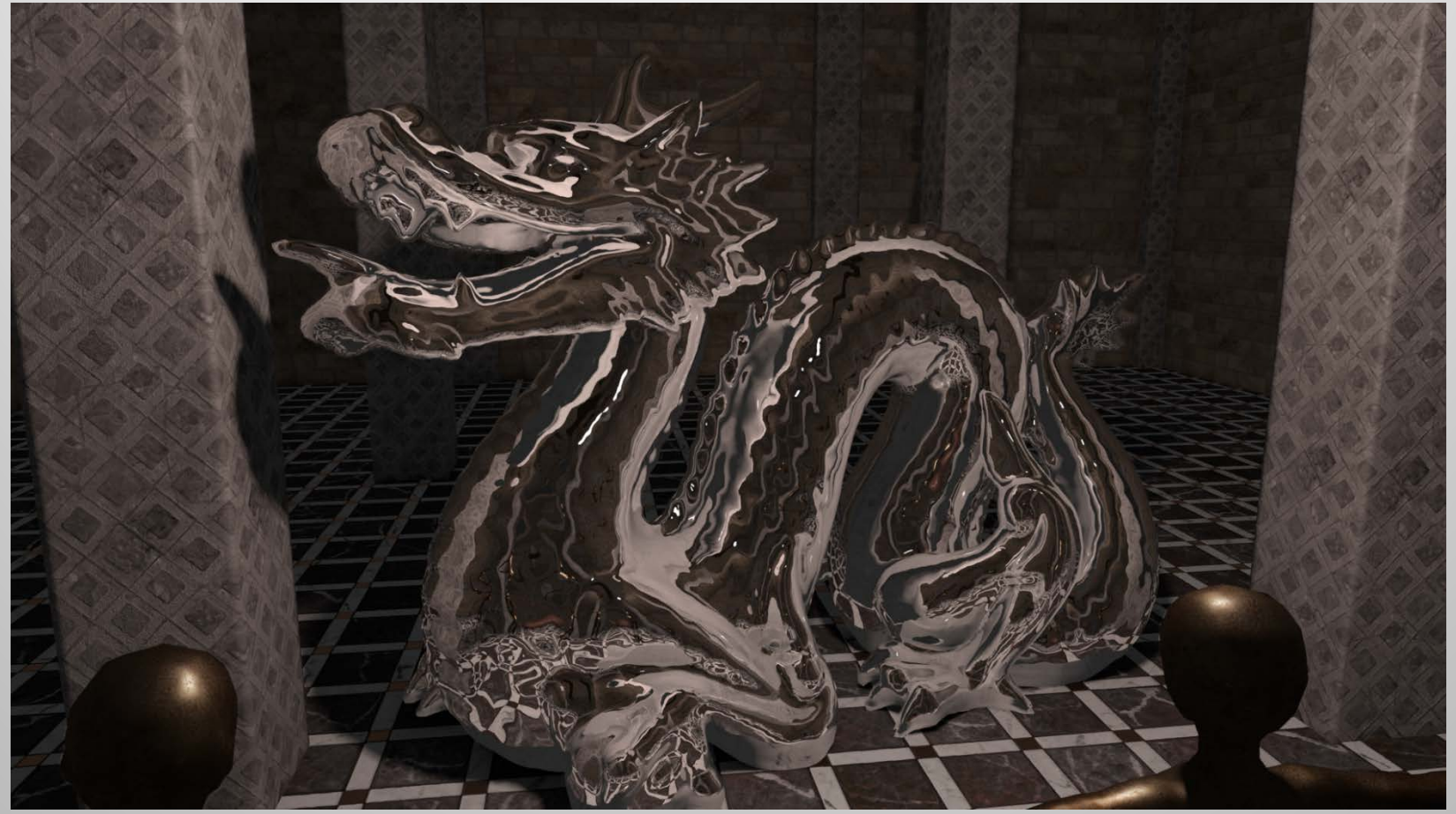
Rendering the World Using a Single Triangle: Efficient Distance Field Rendering

Michael Mroz
Technikum Wien University of Applied Sciences
Visiting at the SIG Center for Computer Graphics, Spring 2017

Goal



Agenda

- What are Distance fields?
- What are they good for?
- What can we do to make them faster?

Agenda

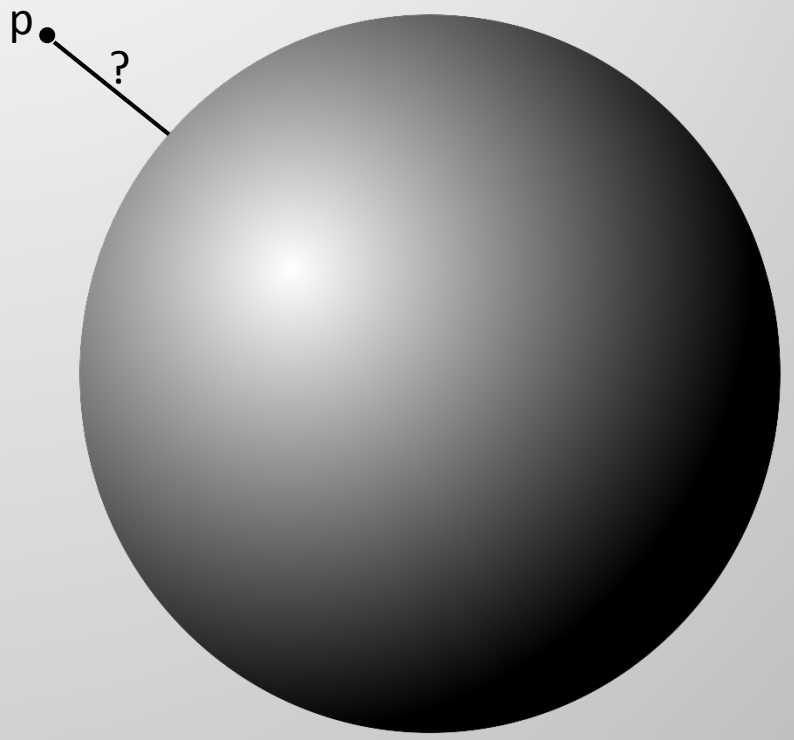
- Distance fields in general
- Rendering with distance fields.
- Optimizations
  - Culling
  - World Distance Field
  - Rasterized Spheretracing

What are Distance Fields?

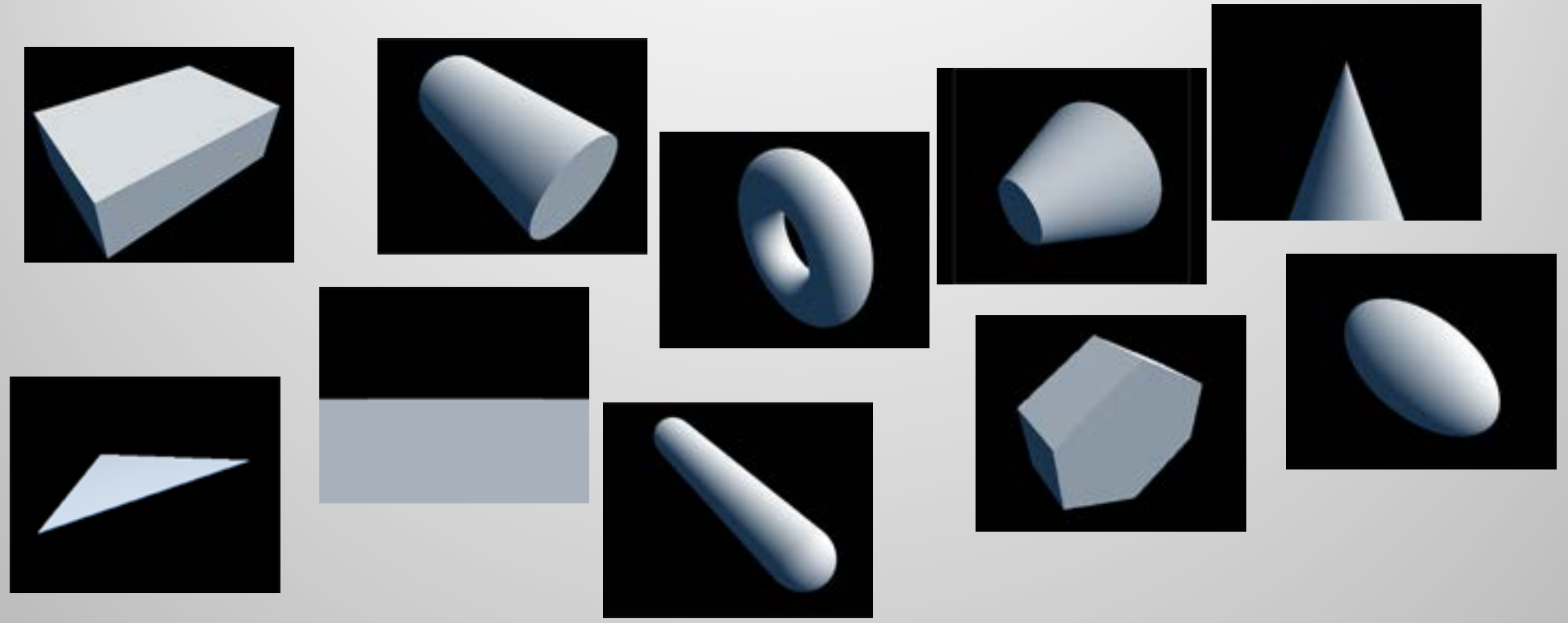
```
CoffeeMugDistance (vec3 p)
{
float distance = ?
return distance;
}
```

What are Distance Fields?

```
SphereDistance (vec3 p)
{
vec3 center = vec3(0,1,0);
float radius = 3.0;
float distance = length(center-p)-radius;
return distance;
}
```

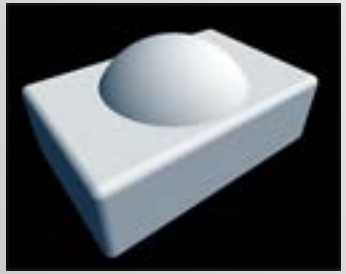


Distance Functions



Images by Inigo Quilez (http://iquilezles.org/www/articles/distfunctions/distfunctions.htm)

Boolean Functions

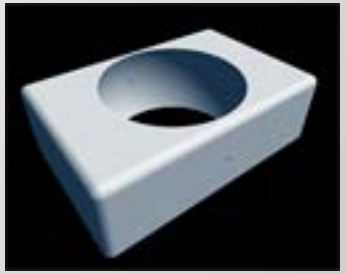


Or

```
float union( float d1, float d2 )
{
   return min(d1,d2);
}
```

And

```
float Intersection( float d1, float d2 )
{
   return max(d1,d2);
}
```



Xor

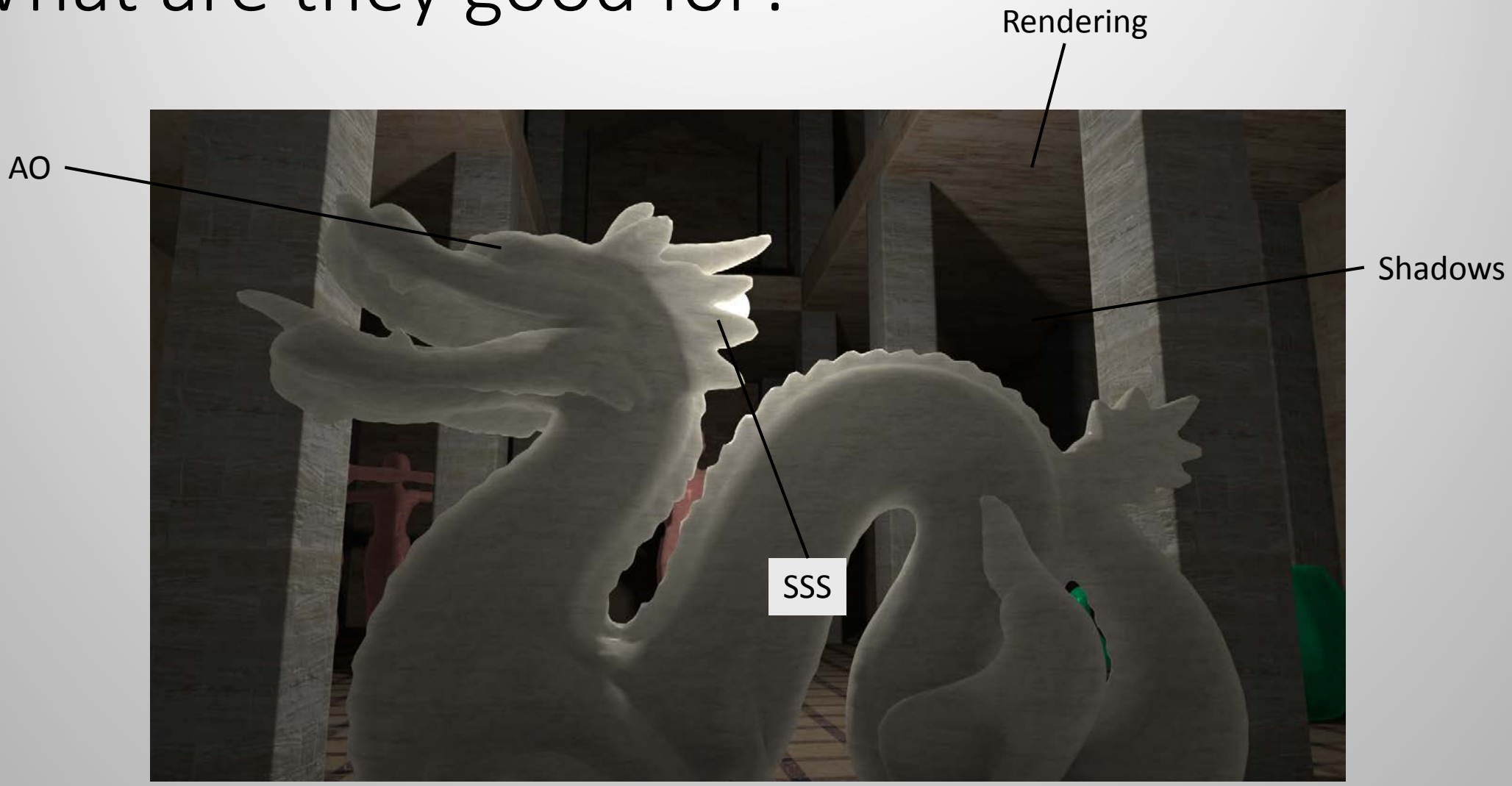
```
float subtraction( float d1, float d2 )
{
   return max(-d1,d2);
}
```

Images by Inigo Quilez (http://iquilezles.org/www/articles/distfunctions/distfunctions.htm)

Coffee Mug?

```
CoffeeMugDistance (vec3 p)
{
float handle = torus(p);
float body = or(handle,cylinder(p));
float distance = xor(body, smallerCylinder(p));
return distance;
}
```

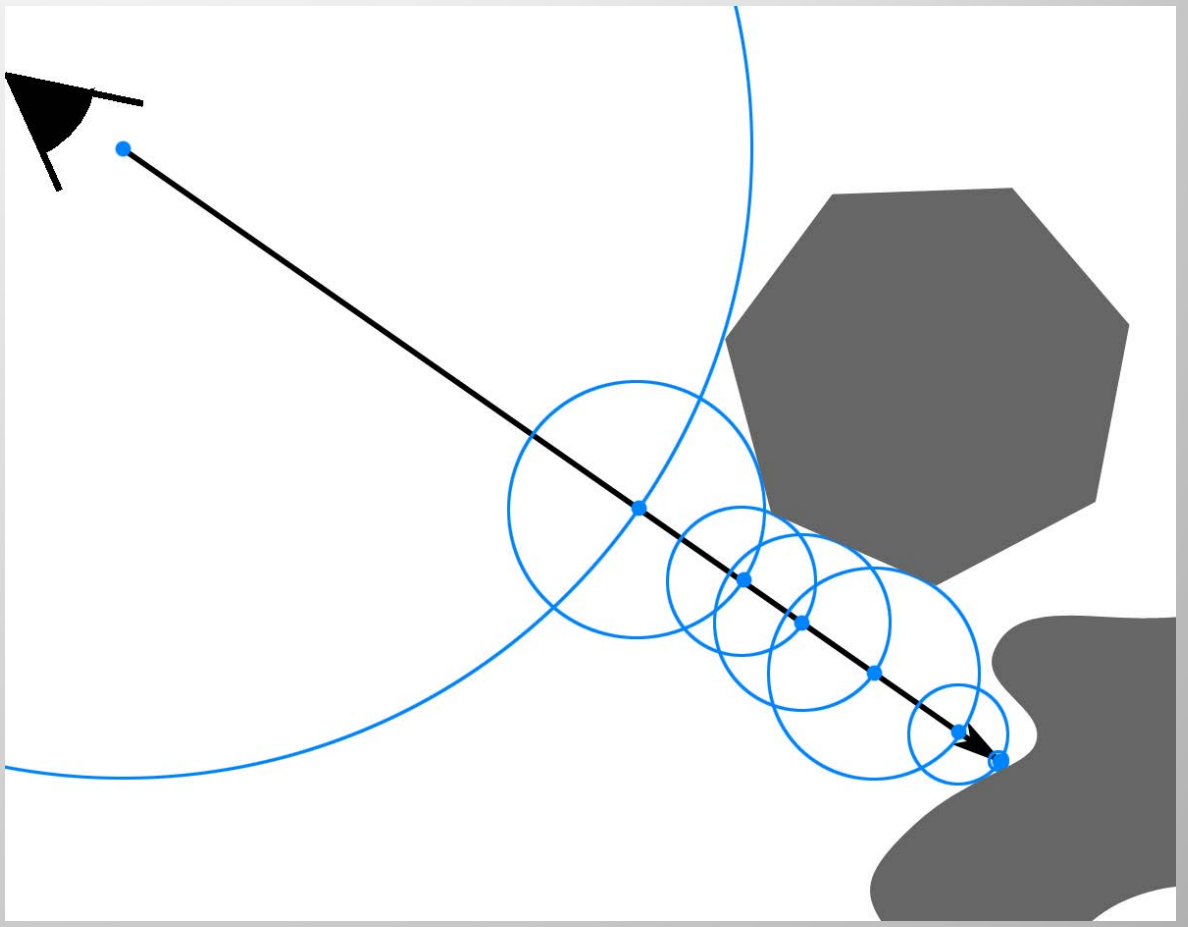
What are they good for?



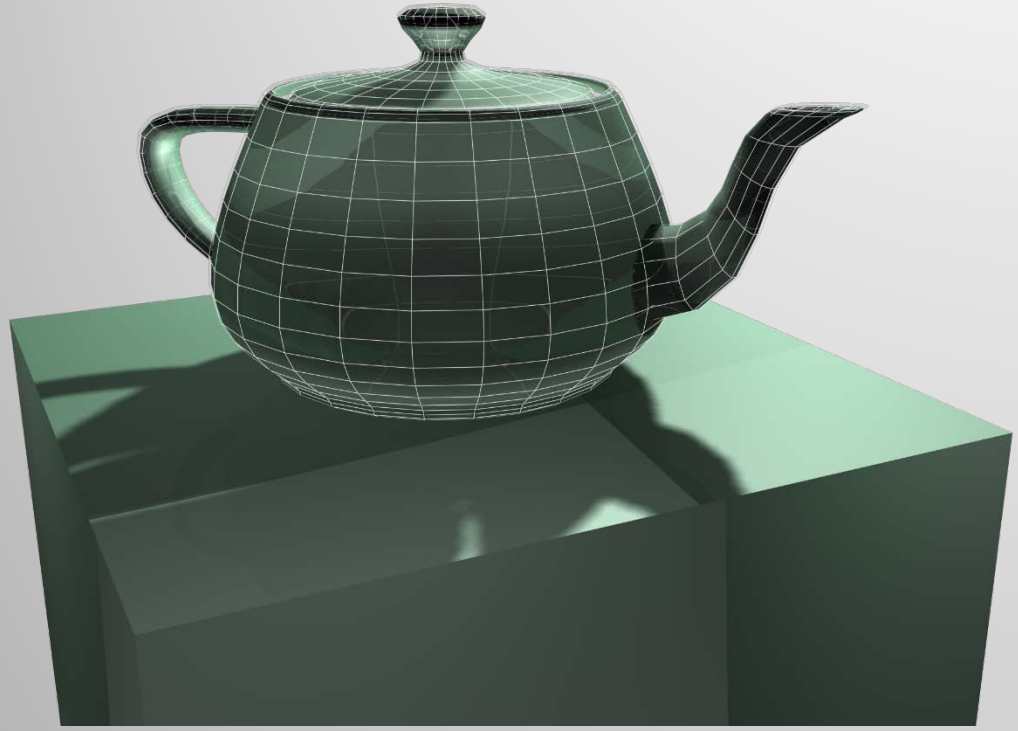
Spheretracing

For each pixel:

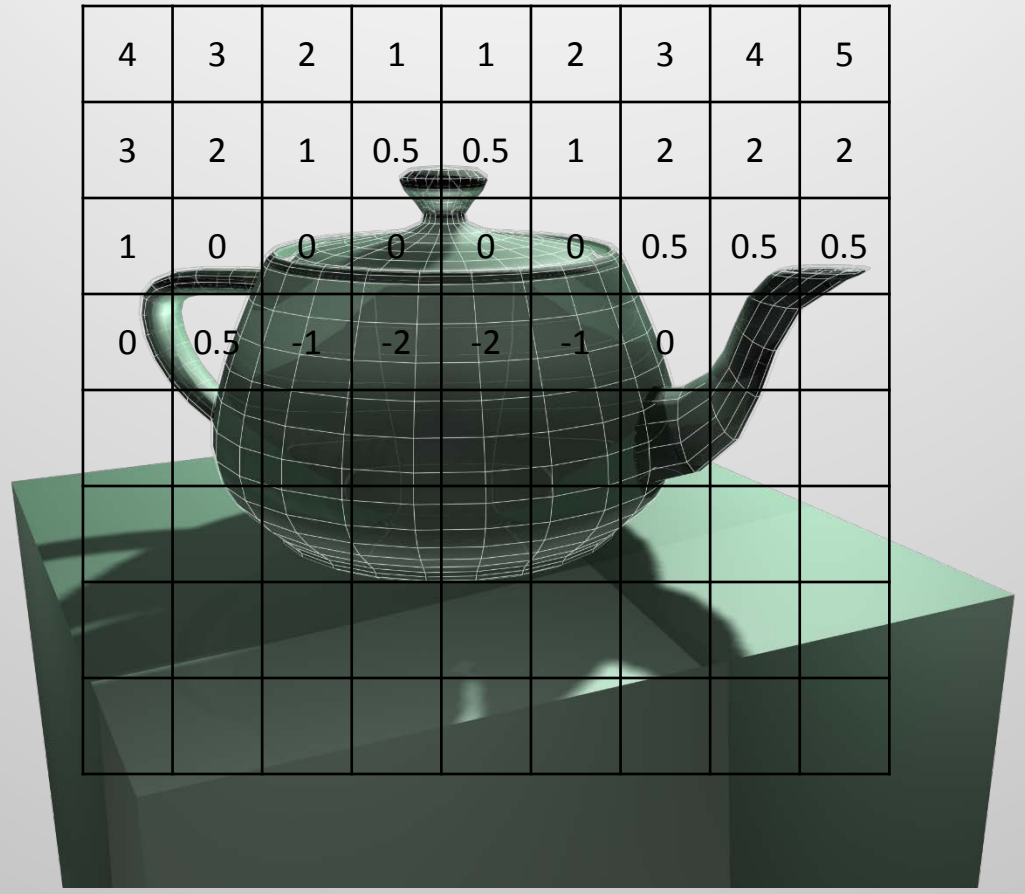
```
samplingPoint=O;
step = SDF(samplingPoint);
while(step > 0)
        step=SDF(samplingPoint);
        samplingPoint += step*D;
```



Distance Transforms



Distance Transforms



How bad can the performance be?

Single Shader

5 Objects	36.37 ms
10 Objects	70.54 ms
15 Objects	134.1 ms
20 Objects	298.52 ms

…URGH!!

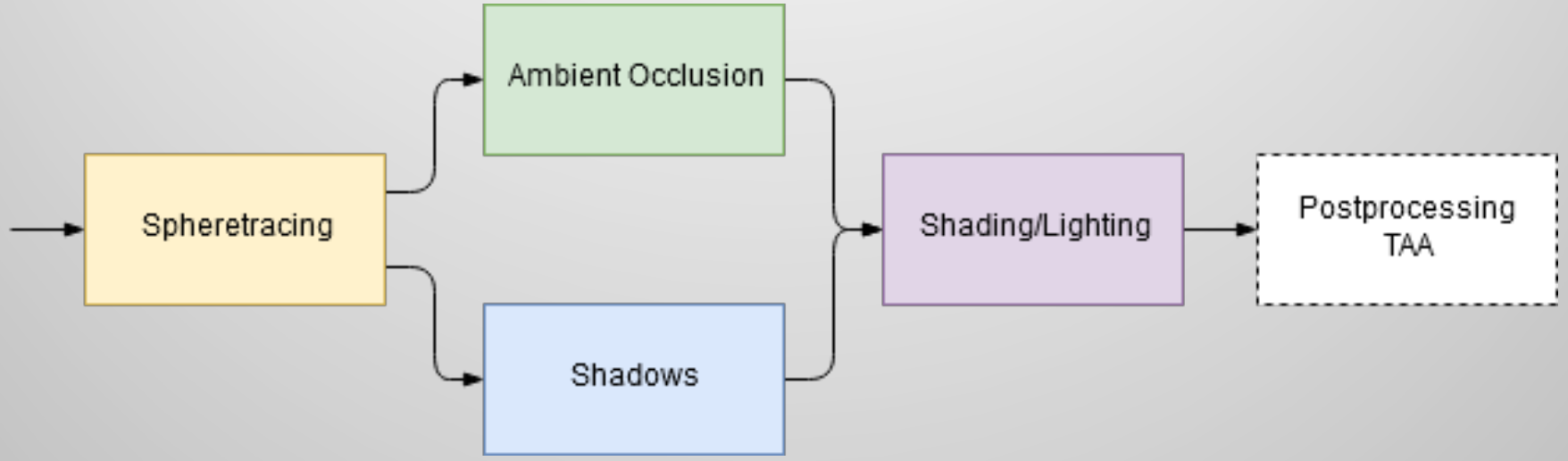
Textures are the enemy

1 Texture read for each:

- Pixel
- Distance Field Texture
- Step on the ray
- Rendering Effect

1920x1080 * 5 * 200 * 4 = ~8.3 bn worst case texture reads per frame

Deferred Rendering



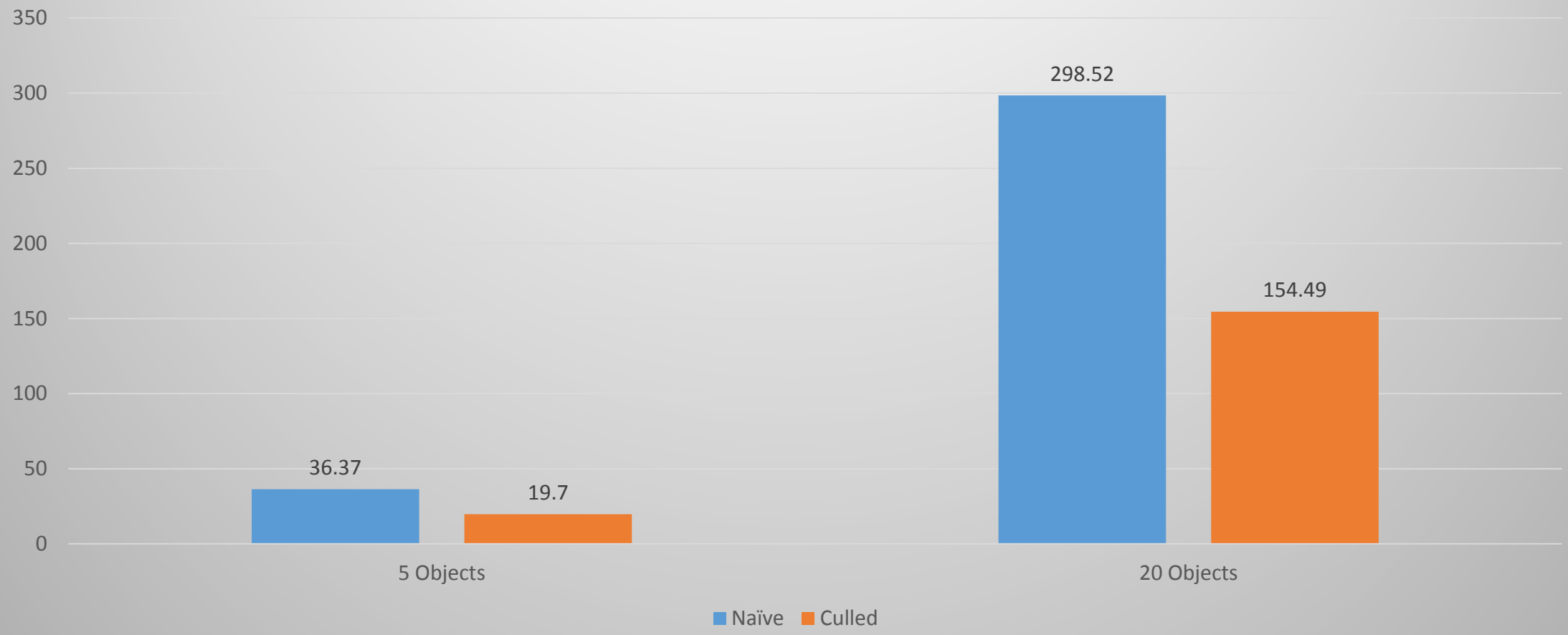
Culling

Run as preprocessing step each frame in 8x8 pixel tiles.

Perform simple ray/sphere, ray/OBB intersection tests and add all positive results to culled list

Only the fields in the culled tile list need to be checked by spheretracing.

Culling Performance

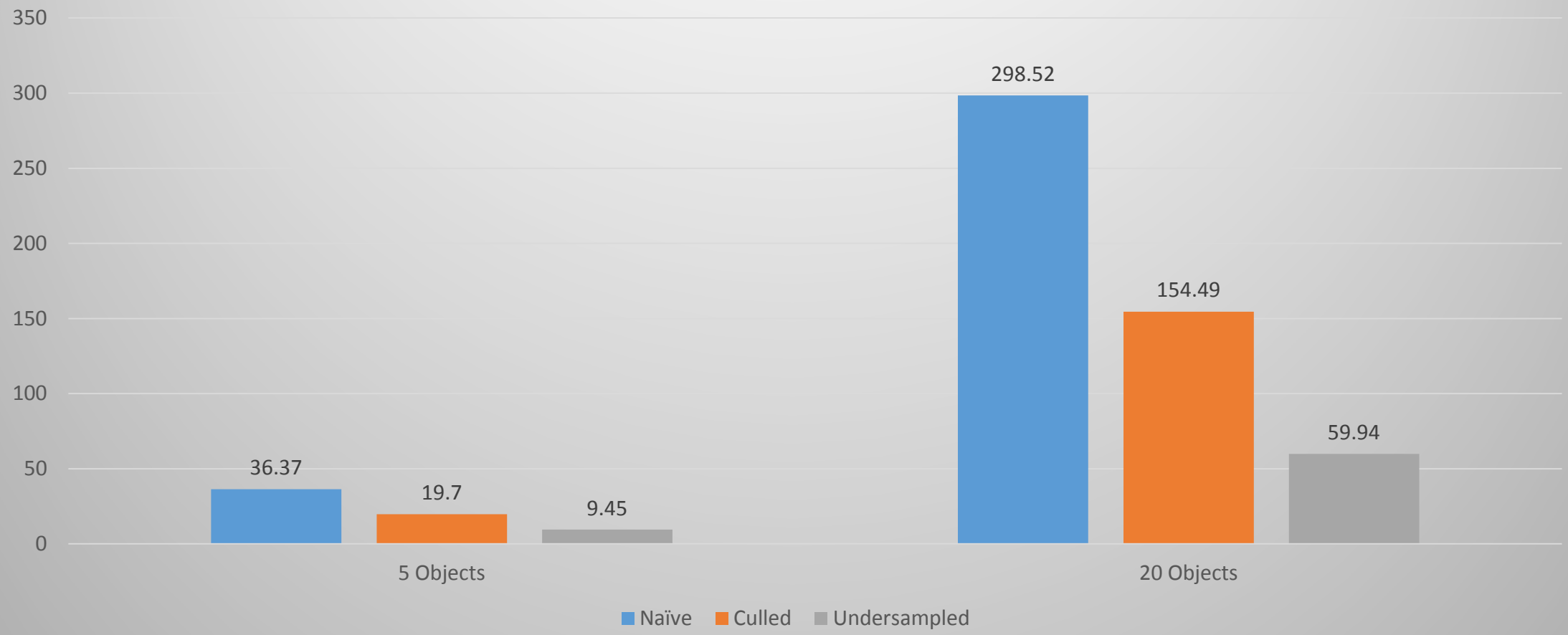


Undersampling

- Run AO & Shadows calculations in lower resolution
- Improve visual quality through bilateral upsampling

Bilinear Filtering	Gaussian Blur	Bilateral Filtering
		

Undersampling Performance



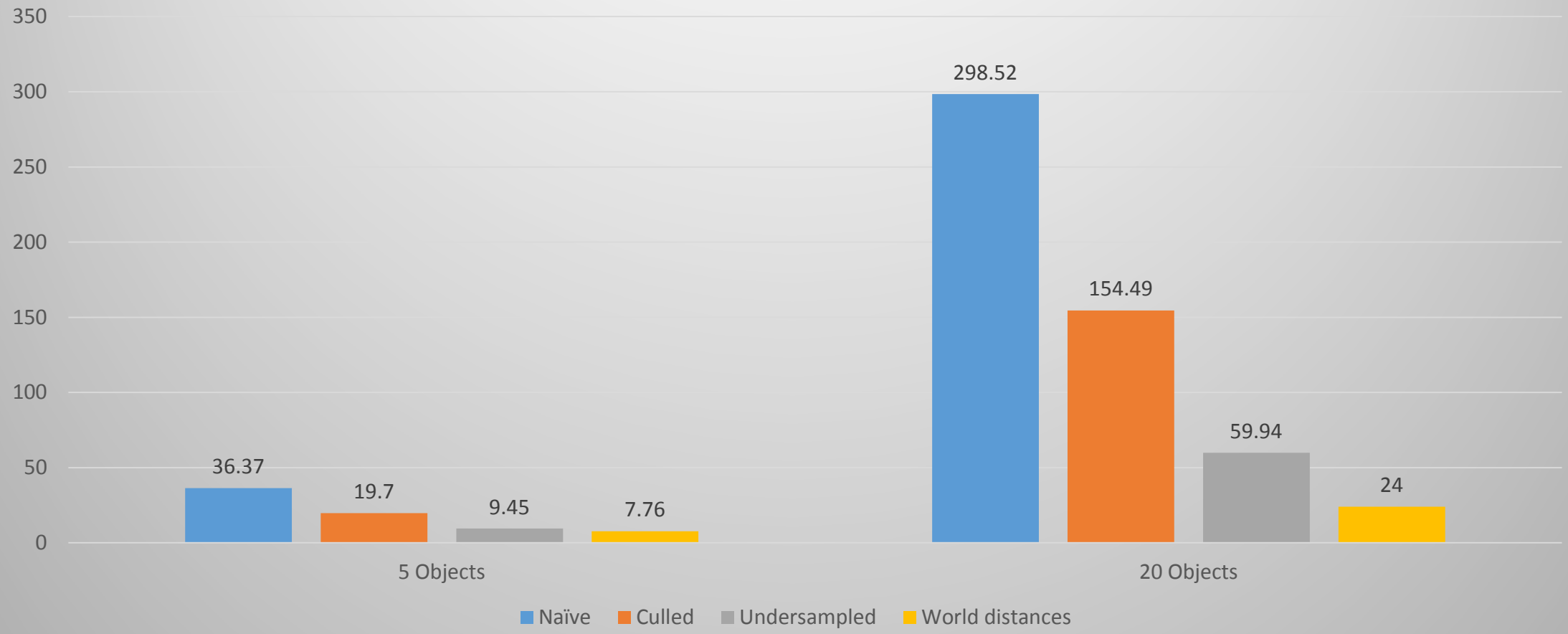
World distance field

One big dynamic distance field that contains conservative estimates of all the distance fields of the scene.

Can be calculated a single time at the start of the program, and then only updated in small portions during run time.

Each time that an algorithm wants to check individual object fields we can check the global field instead, and only jump to individual fields if we are close to an actual object.

World distance field performance



General Optimizations

- Bind meshes tightly
- Use every single bit in your buffers.
- Single Triangle fullscreen

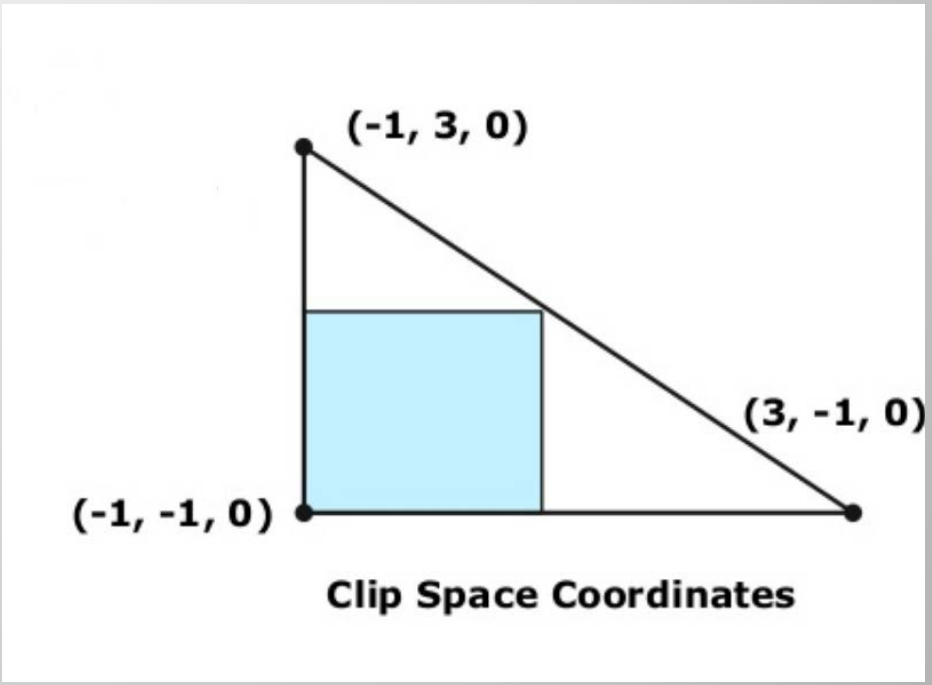
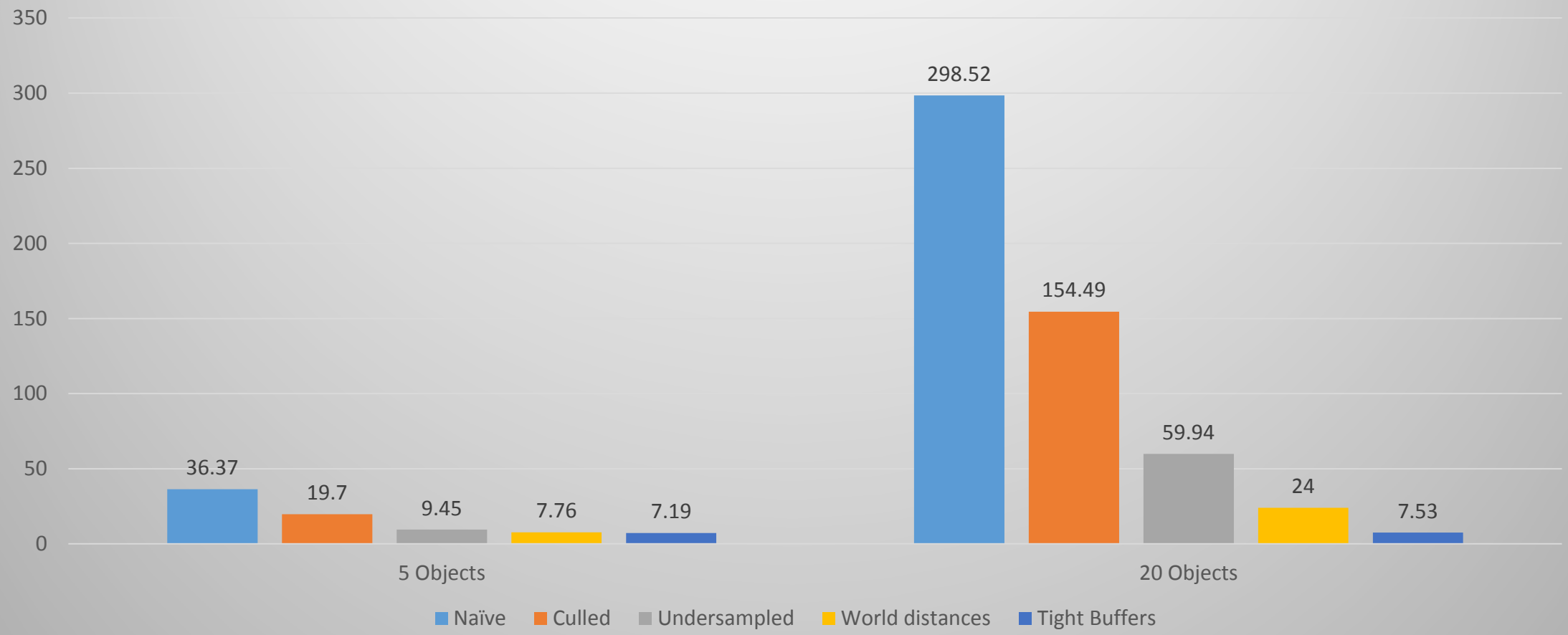
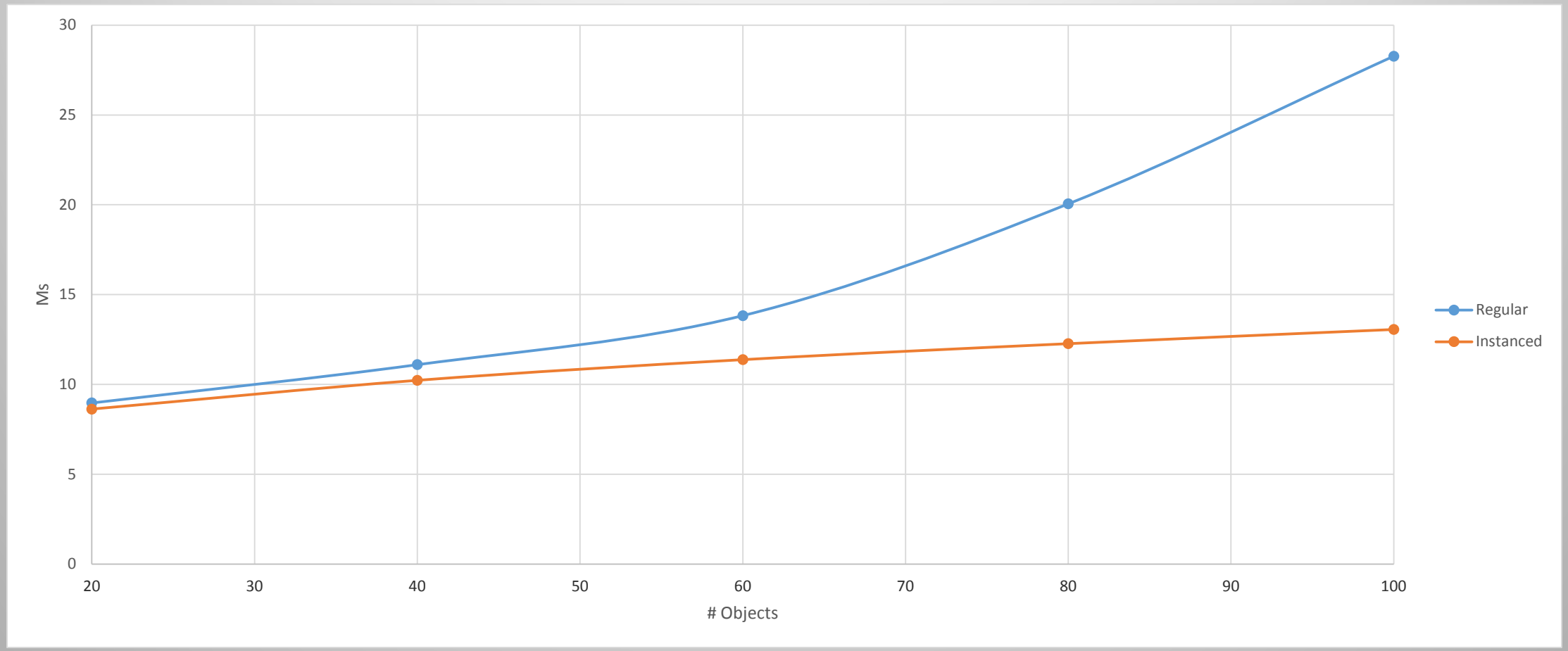


Image by Bill Bilodeau
(https://www.slideshare.net/DevCentralAMD/vertex-shader-tricks-bill-bilodeau)

Final Performance



More Objects?

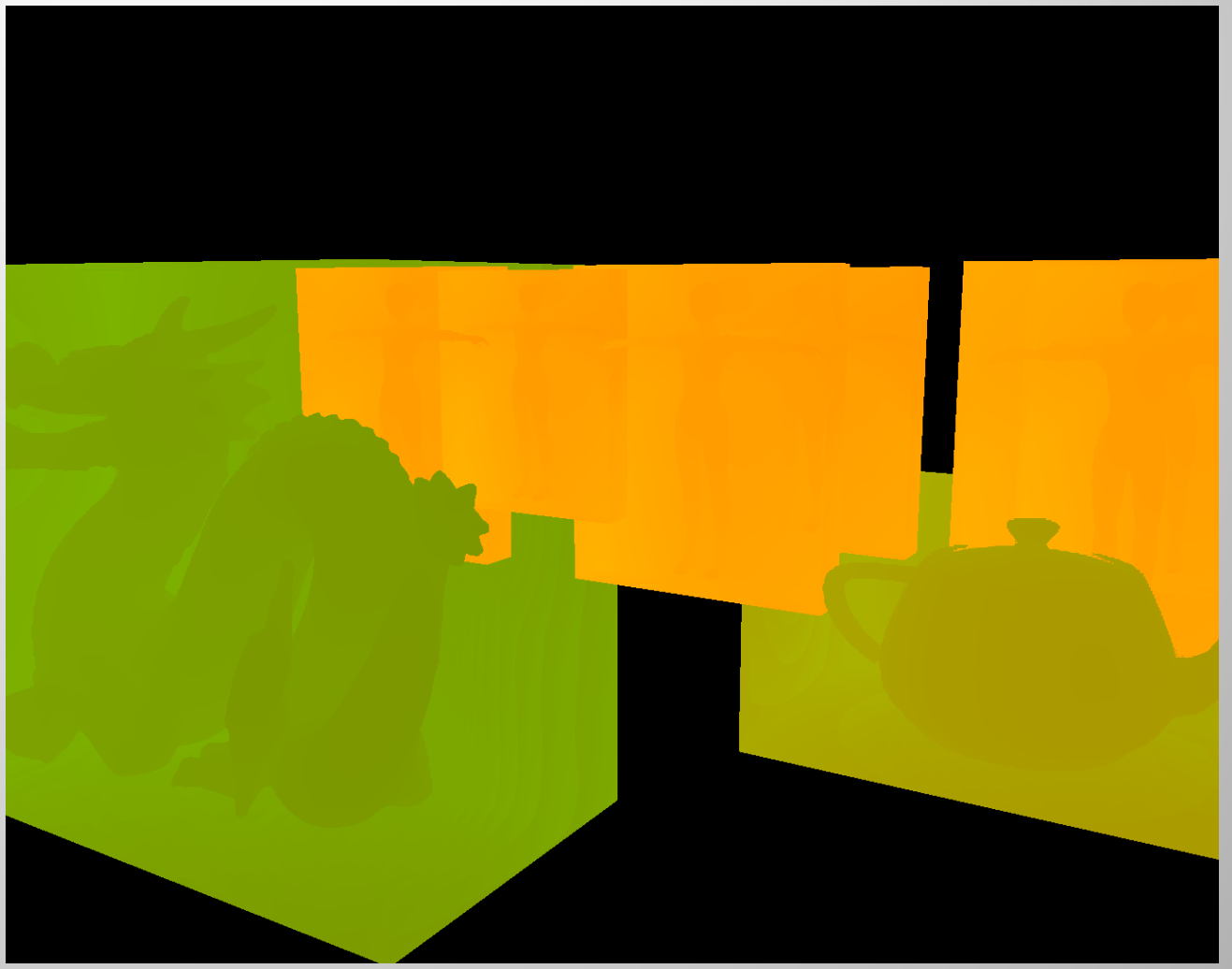


Rasterized Spheretracing

Create distance field bounding boxes as actual geometry.

With each draw call, draw a single box and only spheretrace the object within.

Write the correct depth manually into the zbuffer, and let the depth test reject occluded boxes.



Rasterized Spheretracing

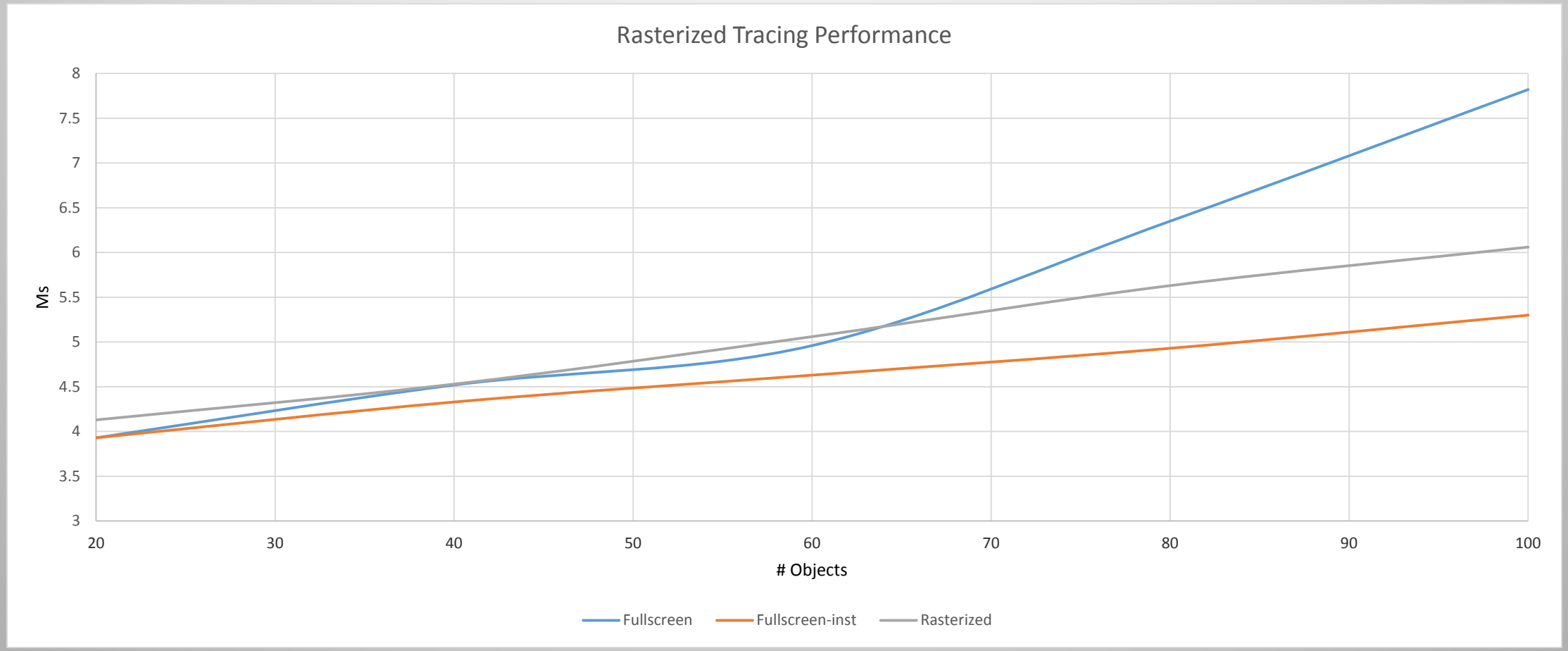
Pros

- Only a single distance field per draw call
- Much fewer steps needed for spheretracing
- Gets rid of some characteristic tracing artifacts
- Fixes our issue!

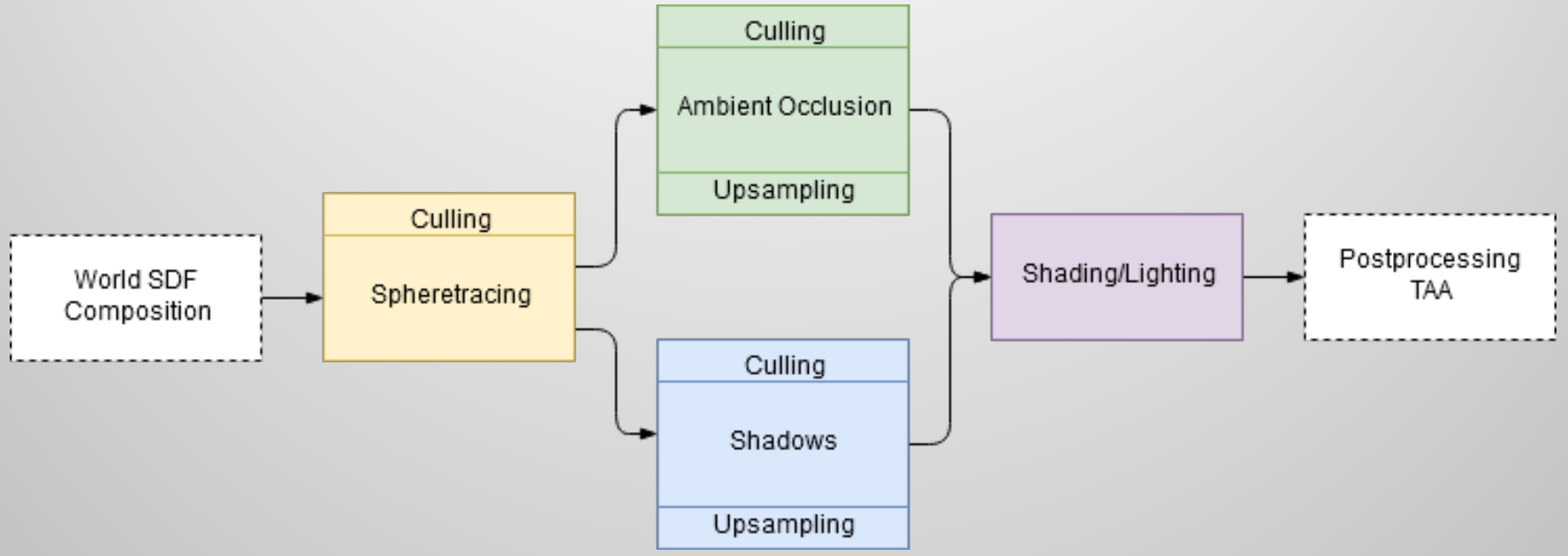
Cons

- A lot of overdraw
- Special case when camera is inside a box

Rasterized Spheretracing Performance



Deferred Distance Field Rendering



The Future

- Distance fields already supported by UE4
- Still high-end feature
- Hot research topic

Closing thoughts

- Whole source code is available online https://github.com/xx3000/mTec

- There are a lot more details and techniques than I could cover. If you are interested don’t hesitate to ask or check out my full thesis in the same repo.

- There are a lot of resources online for getting into distance fields, and nice looking results can be achieved fast. (Inigo Quilez’ Blog, Shadertoy, etc.)

Acknowledgments

- Many of these techniques were inspired by Daniel Wright's SIGGRAPH 2015 presentation "**Dynamic Occlusion with Signed Distance Fields"**.
- Inigo Quilez for the basics and the inspiration for this project.
- Prof Norman Badler, SIG Lab and the University of Pennsylvania for supporting my research.

Thank You!