

# **CS559 Lecture 19-20: More Texture**

---

**Part 4:**

**SkyBoxes and Environment Maps**

# How do we fake lighting?

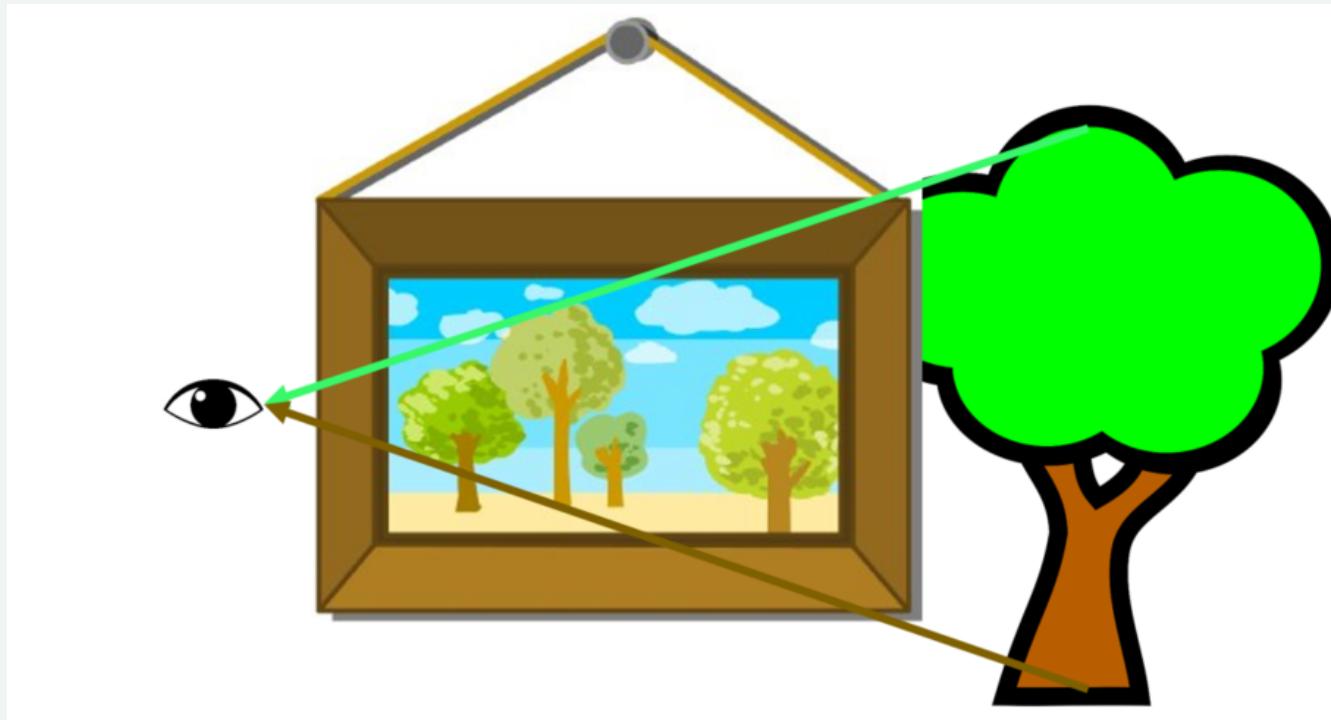
---

- Fake stuff far away
- Fake reflections by pretending things are far away

# Do we have to draw everything?

---

Why not use a (pre-computed) picture?



# Far Away and Not Changing

---

Could draw a box around the viewer

**Positions do not matter**  
**Orientations do**

---

# Sky Box

---

A Box for the "Background"

Always stays centered at the camera

The box moves with the camera)

(beware fake sky boxes)

# In THREE.js

---

Sky Box is built in! (no excuse for fake!)

- Cube Texture (6 images)
- `scene.background` = some cube texture

# How does this help with reflections?

---

Assume the objects being looked at are far away (a Sky Box)

Viewing direction matters

Point Position doesn't (since it is small - objects are far)

# Environment Mapping

---

A Cube or Sphere texture map (around object)

Assume point is at center of Sphere/Cube

Lookup direction is based on viewing direction

# How to store the maps

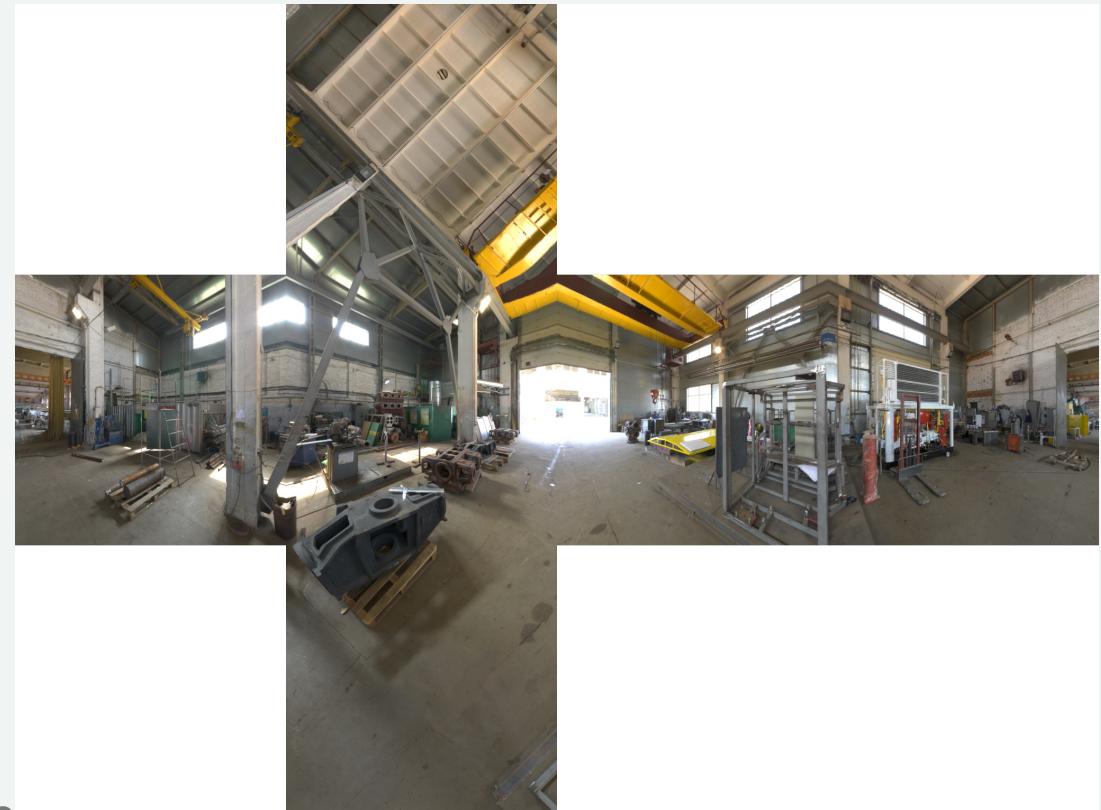
---

## Sphere Maps

(Equi-Rectangular)



## Cube Maps



# Representing the Environment

---

## Sphere Maps

- Sampling issues at pole
- Single Images
- Capture in 1 Photograph

Tools to convert images between forms

## Cube Maps

- Sampling Issues in Corners
- Images are human viewable
- Maps nicely to graphics hardware

# When do Environment Maps Work?

---

Require assumptions:

- Small Object / Far Environment
- Eye is far (only direction matters)
- Position doesn't matter

Small curvy objects - good

Large flat objects - bad



# Very easy in THREE

---

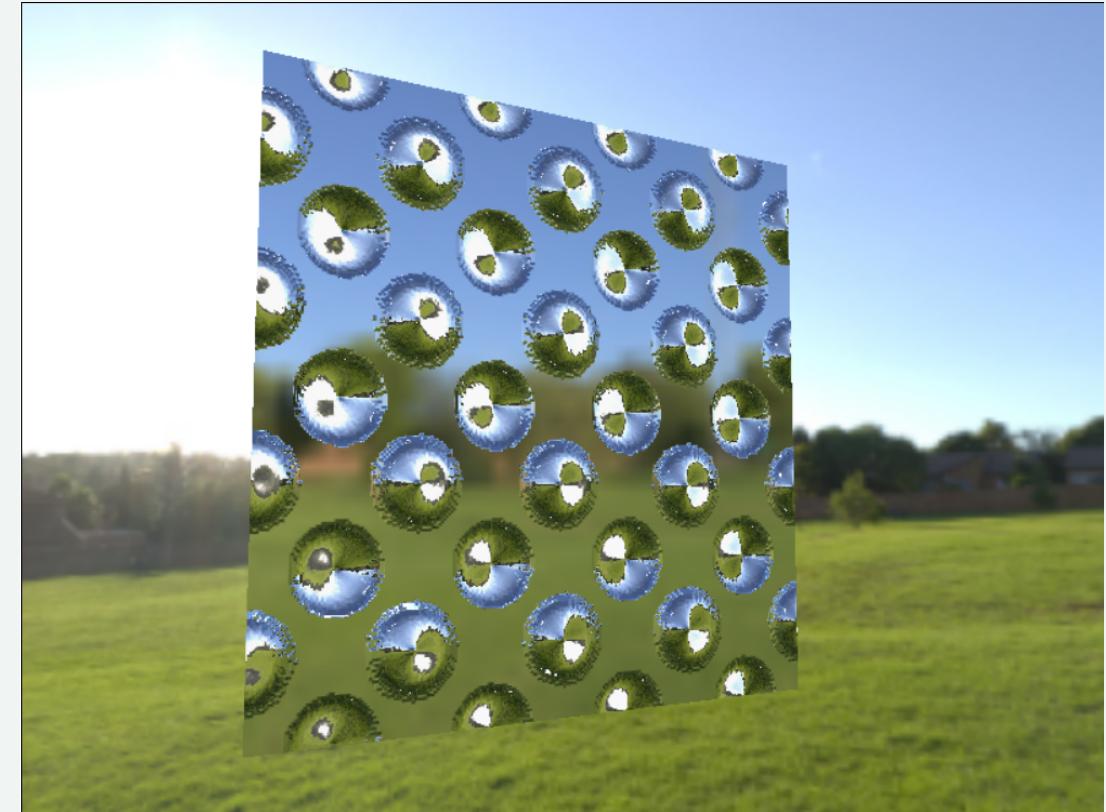
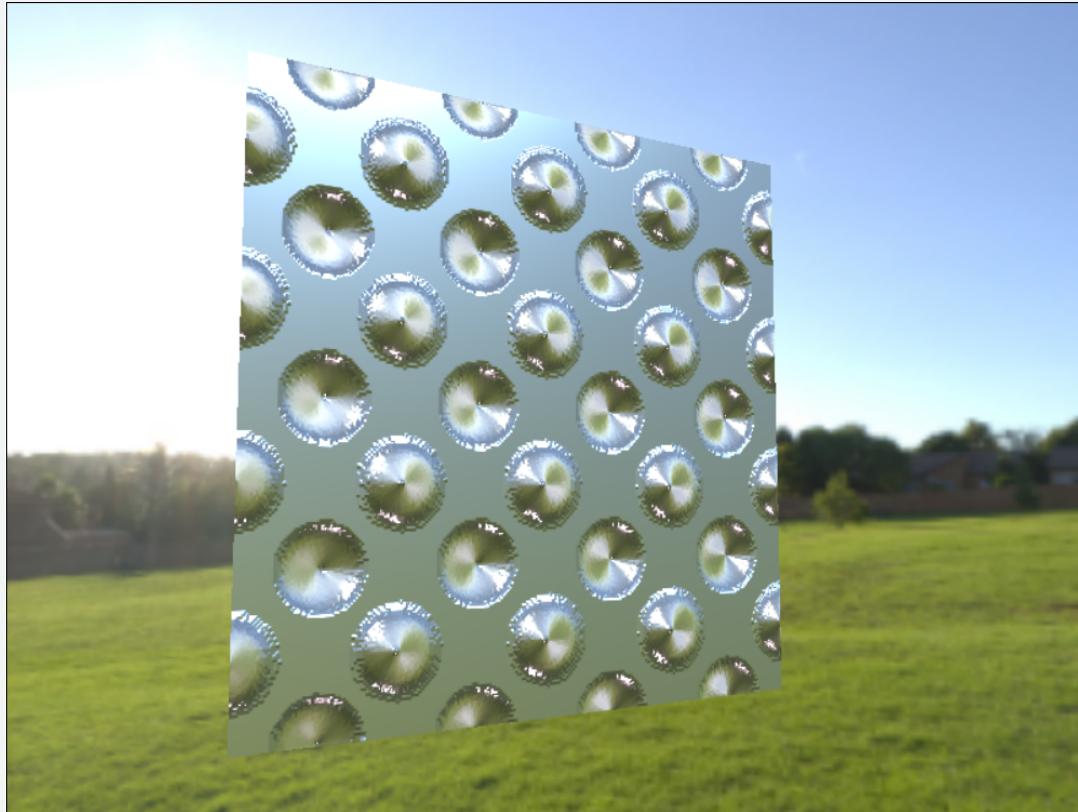
```
// assuming that cubeTexture is a Cubic Texture Map  
let mat = new T.MeshBasicMaterial({ envMap: cubeTexture });
```

A lot goes on...

- U,V computed from view direction
- Lookup into cube texture map

# Works well with bump maps!

---



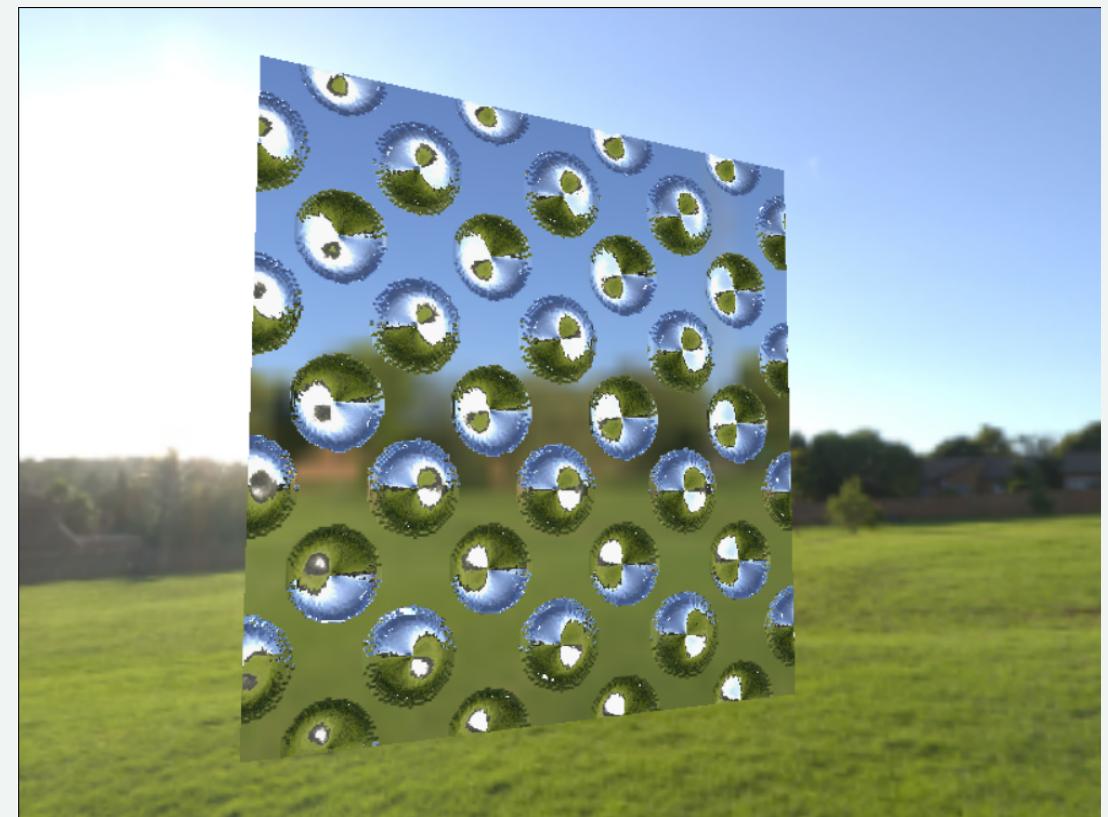
# Requires a Shiny Material

---

Using THREE.js `MeshStandardMaterial`

- metalness 1.0
- Roughness 0.0

(try the demo)



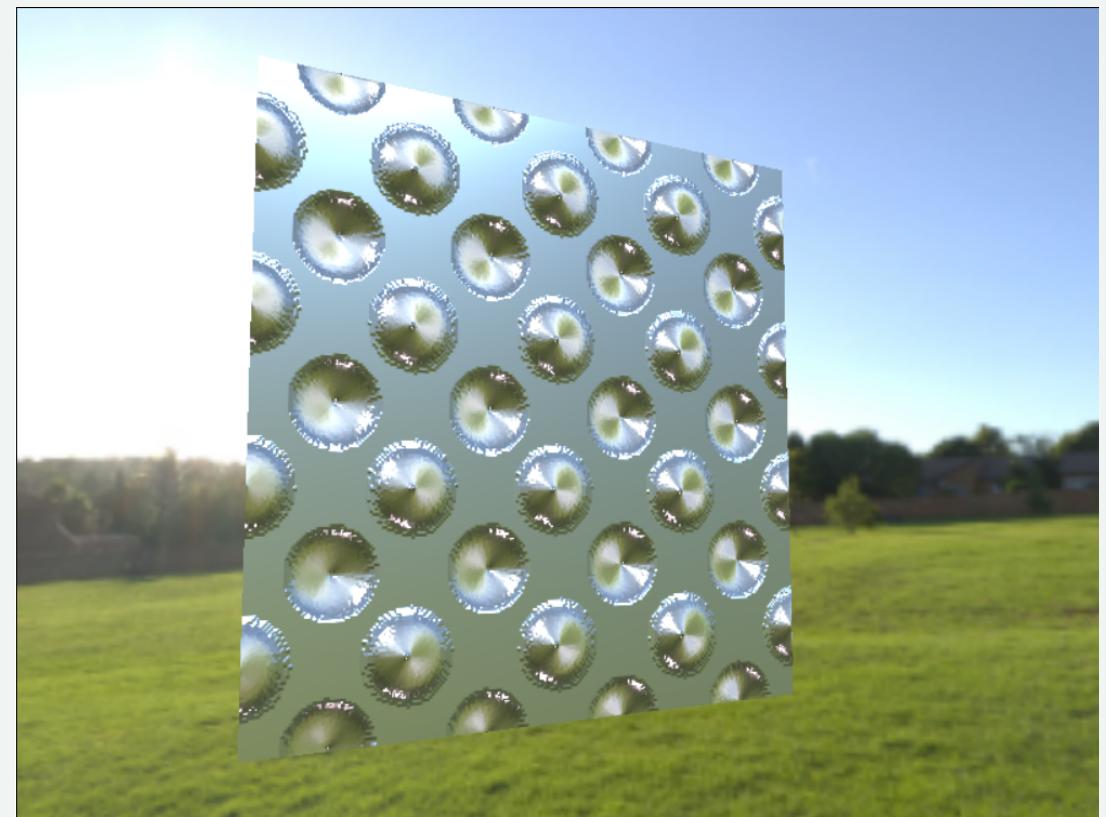
# Less "reflective"

---

How much to mix in "regular lighting"

How much sharp are reflections

Rough materials start to look **diffuse**



# Environment Maps for Lighting

---

The environment map is light!

Use the real scene to create brightness!

Not limited to a small set of light sources!

# But the environment map is static?

---

Often, pre-compute the environment map ahead of time  
(or use a photo)

But: we can draw it ourselves!

- draw the scene with the camera where the object will be
- use a special "cube camera" to take a picture in 6 directions
- take this picture before making the "real" picture
- use this picture as the texture
- **Dynamic Environment Map**

Multi-pass rendering (draw the scene multiple times)