Stylized Rendering

Screen Space Texture Mapping + Non-Photorealistic Rendering

Yoonje Lee | Daniel Koo

Screen Space Texture Mapping

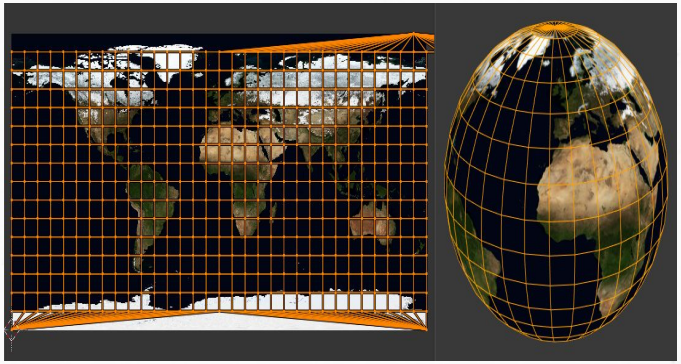
- Texture will “stick” to the screen rather then the 3D object
- This is similar to looking out the window

Methodology

1. Convert vertices from world space to camera space, then camera space to screen space
    - Allows views to be relative to camera's position and orientation
2. Compute barycentric weights on current pixel and interpolate with camera space
3. Convert these values into screen space pixel coordinates by multiplying with screen width and height; this will be the new texture_color for each pixel ($0.0 \leq x,y \leq 1.0$)
4. Use these interpolated value (x,y) to map on to the texture's u, v index
5. Apply shading to the new texture_color in order to give 3D cues

UV Texture mapping vs Screen Space Mapping

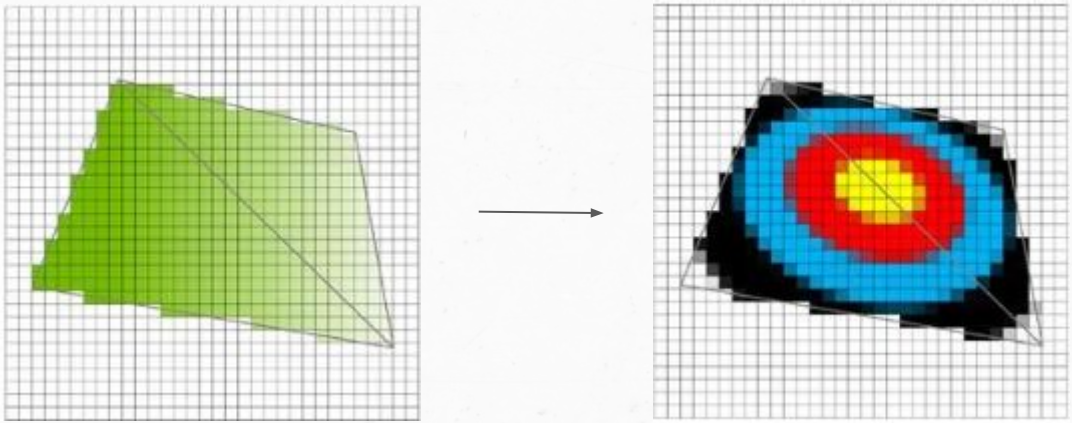
UV Texture Mapping



2D image to a 3D model’s surface

Screen Space Mapping

Texture is applied after the 3D object has been projected to screen as 2D



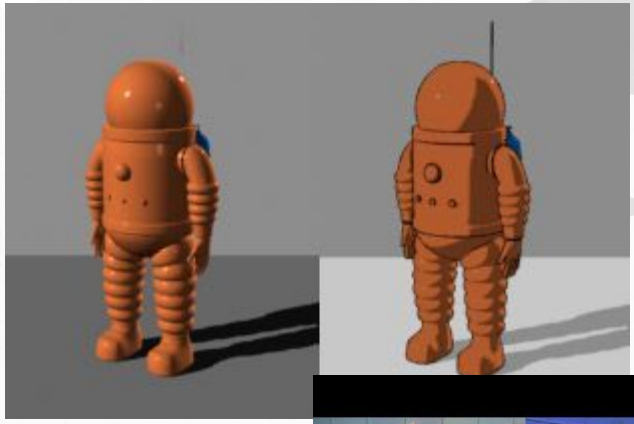
3D object in 2D Screen

Output Without Shading

Output With Shading

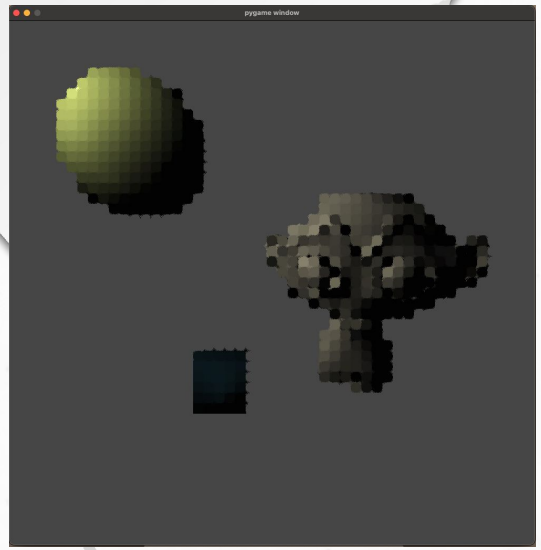
Non-Photorealistic Rendering (NPR)

a computer graphics technique that uses digital art and other technologies to create expressive styles for digital art



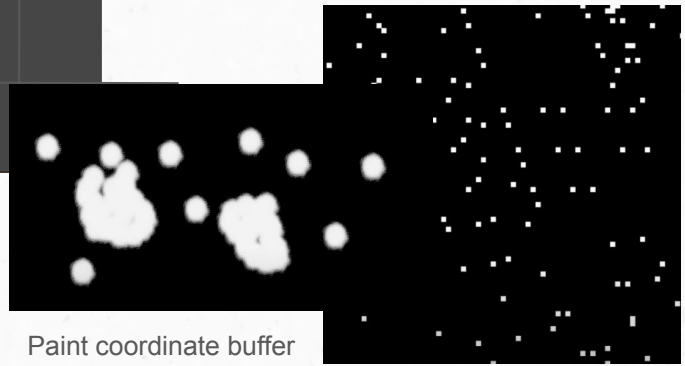
Step 1: Render Brush

Render brush shape



Render Randomly

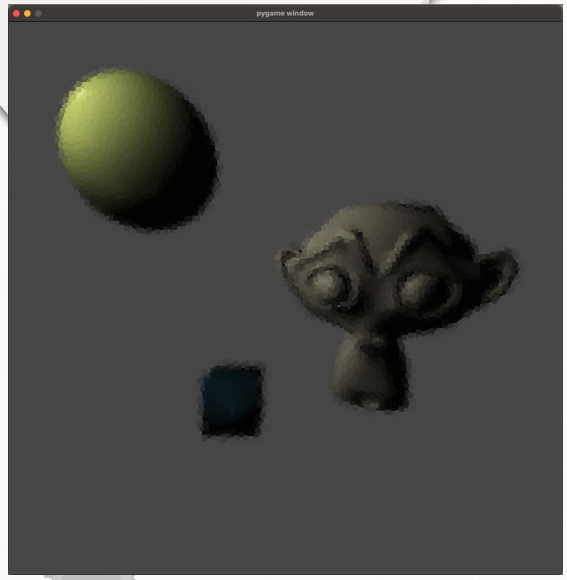
1. Get 10 random pixels from each 100x100 box and render 10 brush strokes
2. Repeat until it fills 98%
3. Color: color of pixel area



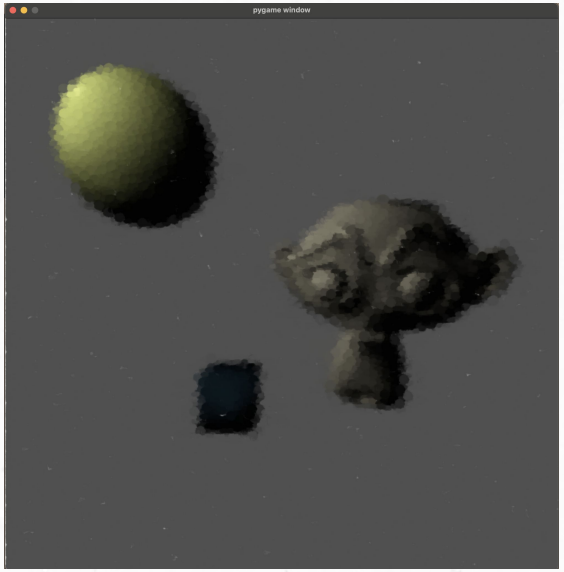
Paint coordinate buffer

Step 2: Brush Type

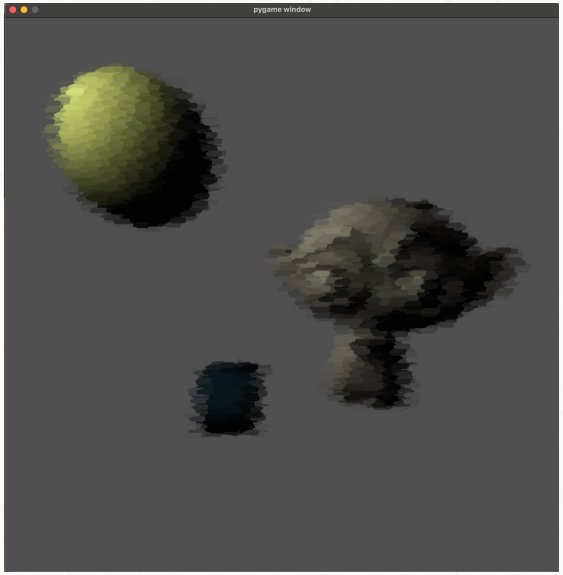
Smaller point brush



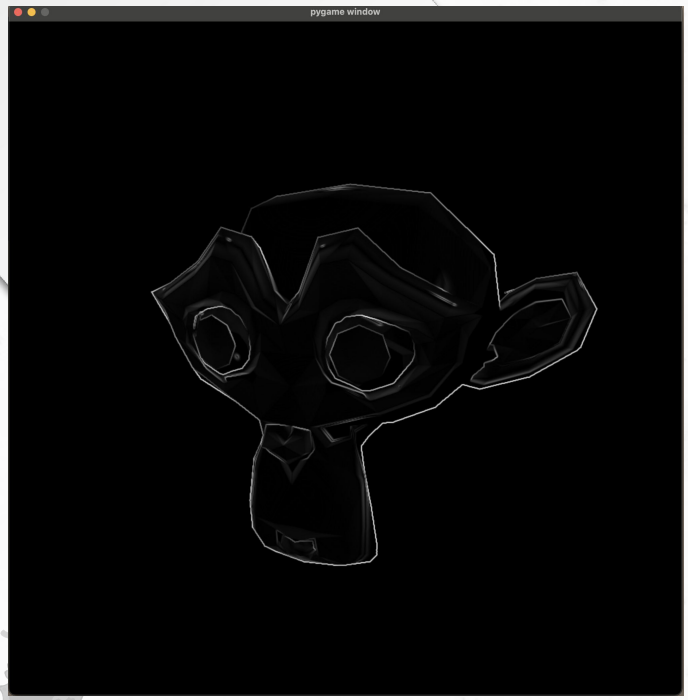
Random size brush



Long stroke brush



Step 3: Sobel filter

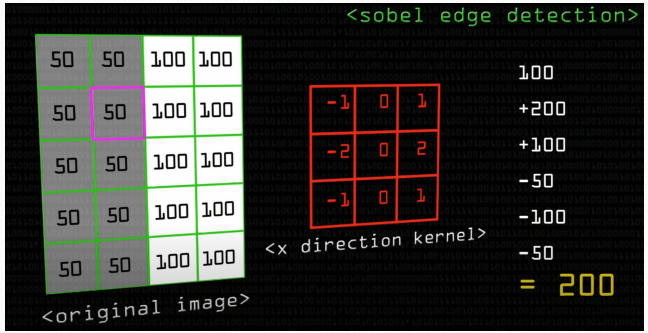


1. Used for edge detection of image.
2. Uses a Isotropic 3x3 Image Gradient operator

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A}$$

$$\mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

Define A as source image



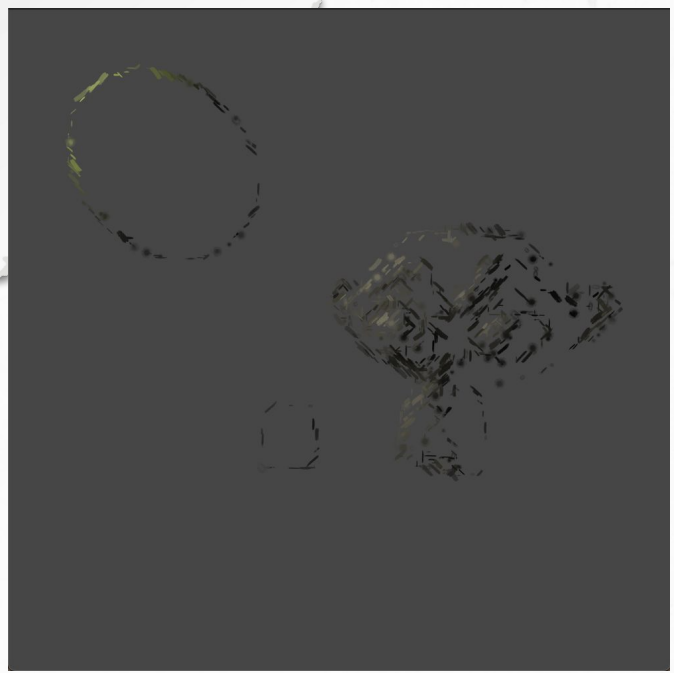
$$\mathbf{G} = \sqrt{\mathbf{G}_x^{\,2} + \mathbf{G}_y^{\,2}}$$

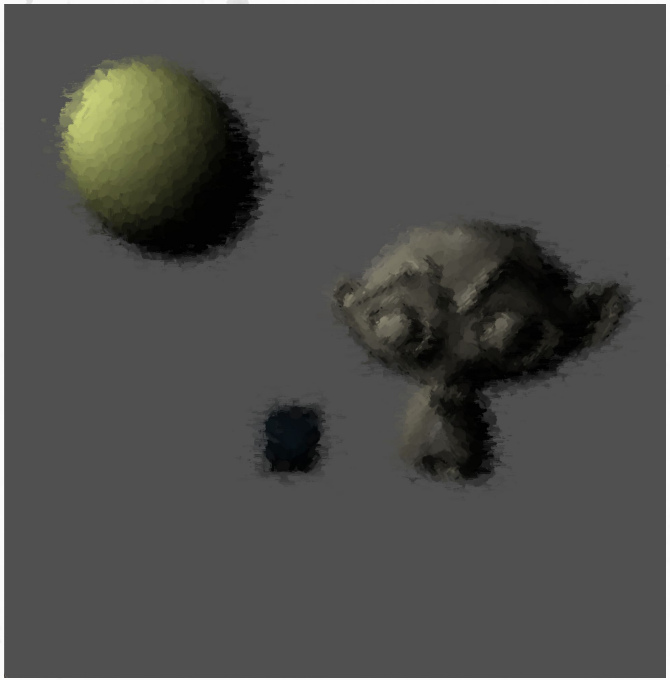
Magnitude

$$\mathbf{\Theta} = \operatorname{atan2}(\mathbf{G}_y, \mathbf{G}_x)$$

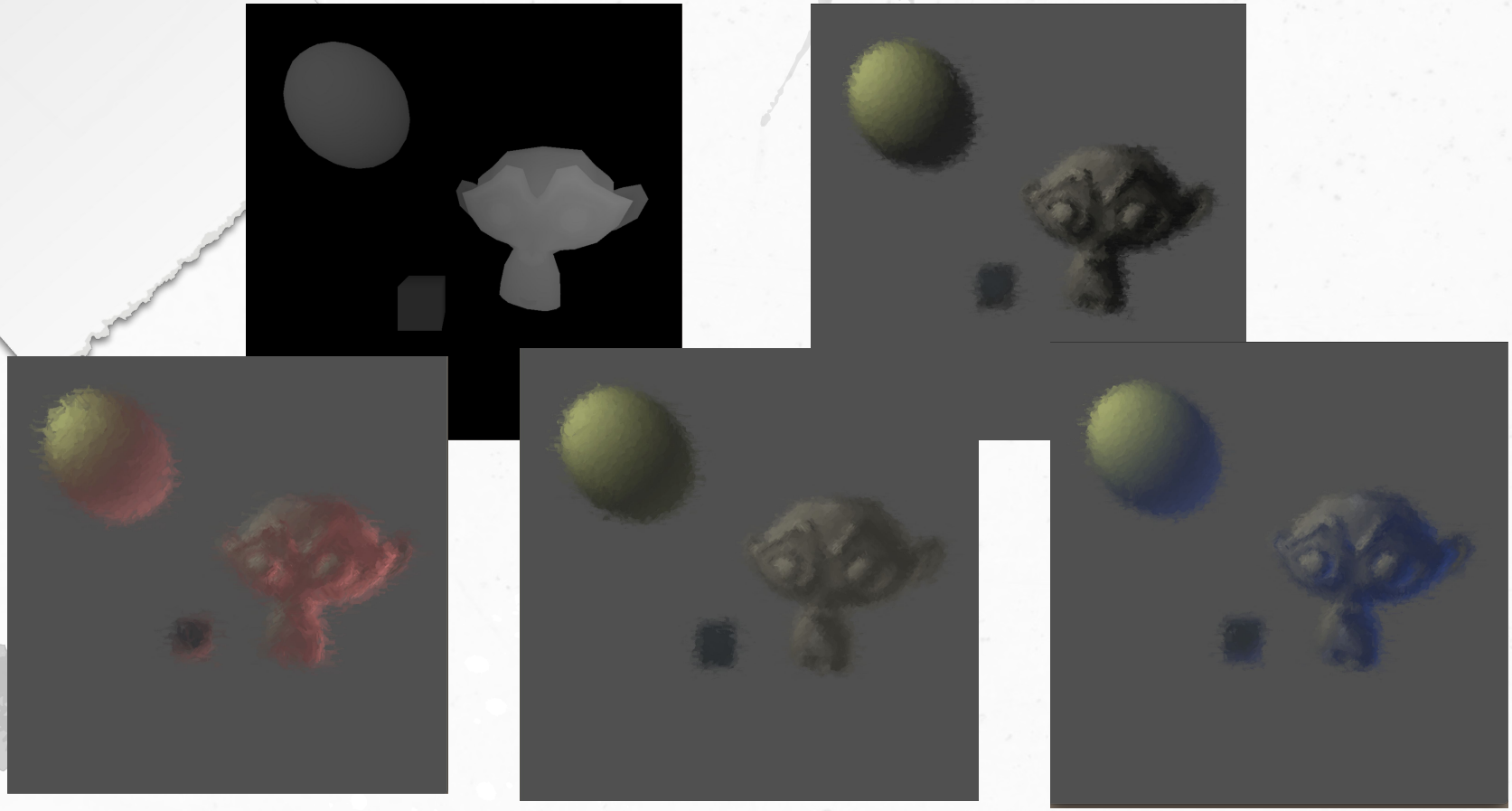
Direction

Step 4: Apply Sobel filter





Step 4: Experiment



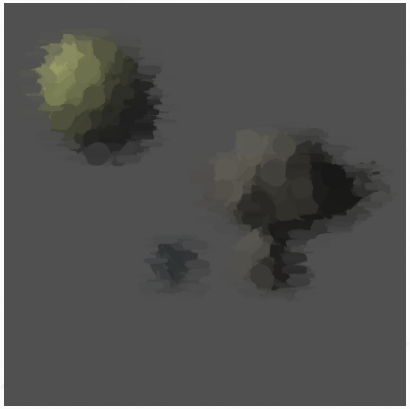
Methodology

1. Load brushes
    a. Size: 60, 40, 20, 10
    b. Angle 0 to 345 (range(0, 360, 15)
2. While rendering triangle
    a. Buffer (phong)
    b. Faded (depth)
3. Layered painting
    a. Paint: 60 (98%) -> 40 (80%) -> 20 (50%)
        i. Pick 10 random pixels from 100x100 box
        ii. Paint until filled __ %
    b. Gradient Painting Edges: 20 (98%) -> 10 (97%)

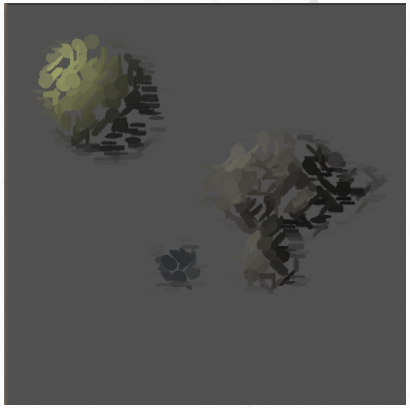
Layer

Paint

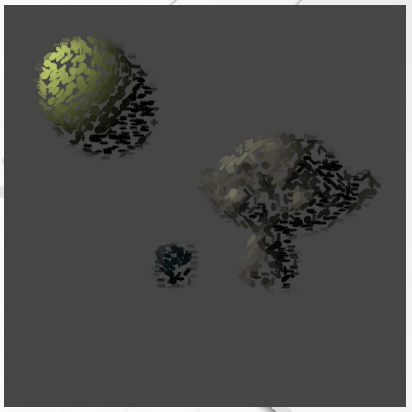
60 (98%)



40 (80%)

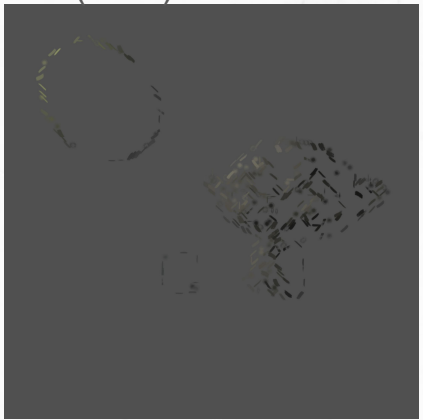


20 (50%)

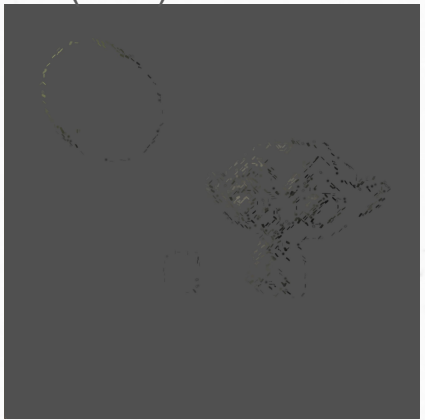


Gradient

20 (98%)



20 (97%)



Combined

Non photorealistic rendering happens after all rendering was completed. (painting on top)

After the screen space texture mapping was completed (including other buffers), we start the painting process. The original image buffer was used to calculate color and angle.