Introduction to Computer Graphics (CS360A)

Instructor: Soumya Dutta

Department of Computer Science and Engineering

Indian Institute of Technology Kanpur (IITK)

email: soumyad@cse.iitk.ac.in

Quiz on October 19th

- We will have a quiz on October 19th during the regular class hours
- Quiz will have 10% course grade
- Syllabus: Everything up to 16th October's lecture
- Questions similar to the midterm examination

Post Processing of Fragments

- Can we use shaders to perform image processing in GPUs in real time?
- Yes, we can!
  - We can render the scene in a framebuffer object and then render the scene again by looking up texture values from framebuffer and apply image processing operations on pixel values
  - We can load an image that we want to process into a texture first and then map it to a quad mesh to show it
    - While showing the texture on the quad, we can manipulate the texture color values and then show it
- These will be done in fragment shader, parallel per fragment, hence blazing fast!

Post Processing of Fragments

- Post processing and screen space operations can be used efficiently to achieve various effects on the output image
- We can build an interactive image processing toolbox that is very fast since it works in GPUs
  - Edge detection, smoothing, sharpening, gray scale conversion, embossing, etc.
  - Adjust brightness, contrast, saturation, etc.

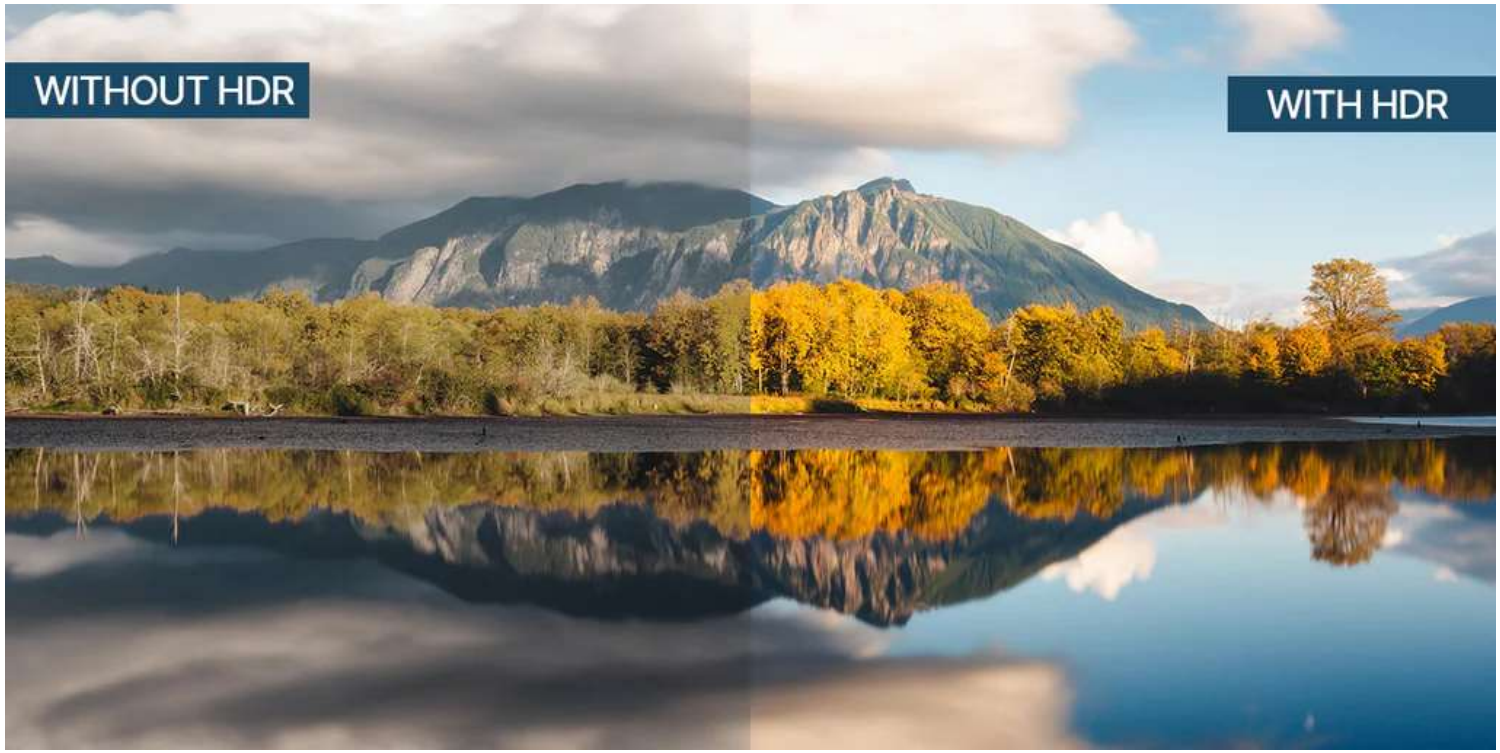
Post Processing of Fragments

- More advanced effects can be added

Bloom effect

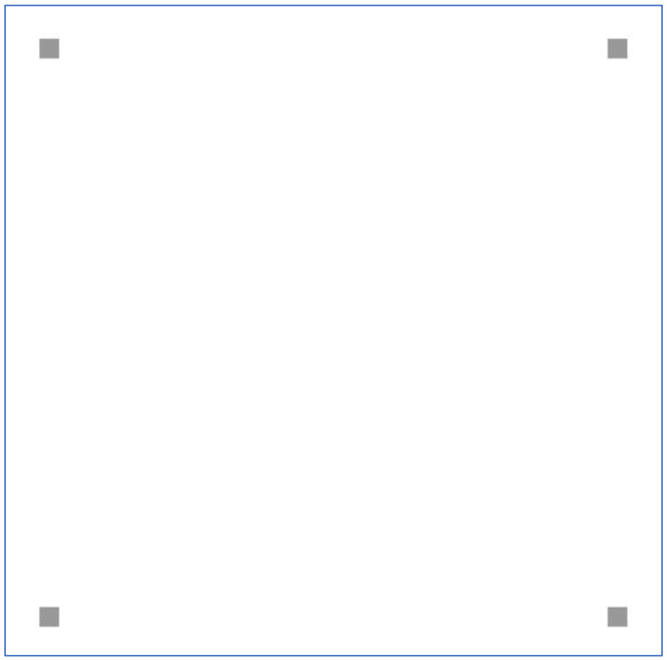
Post Processing of Fragments

- More advanced effects can be added

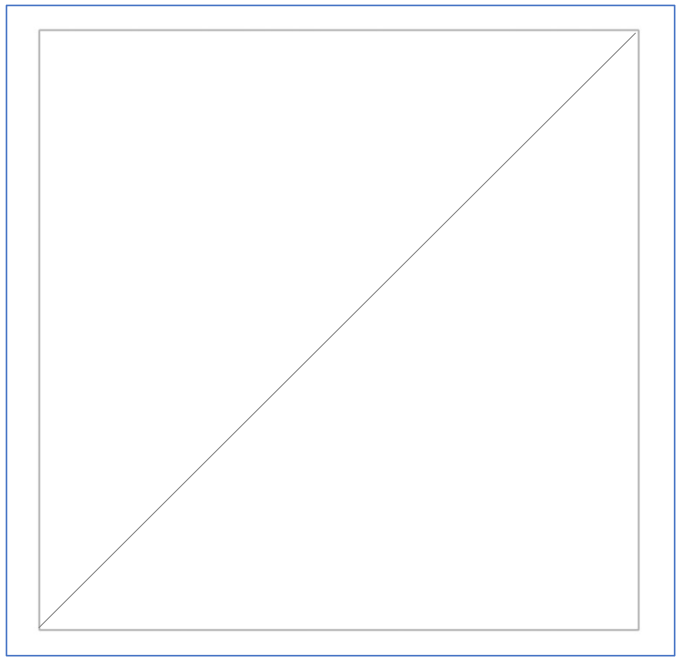


HDR effect

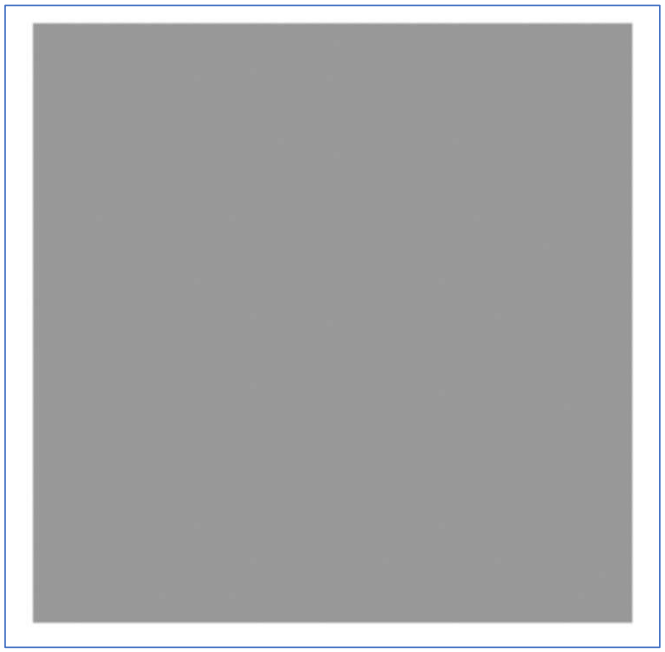
Post Processing of Fragments: Pipeline



4 Points

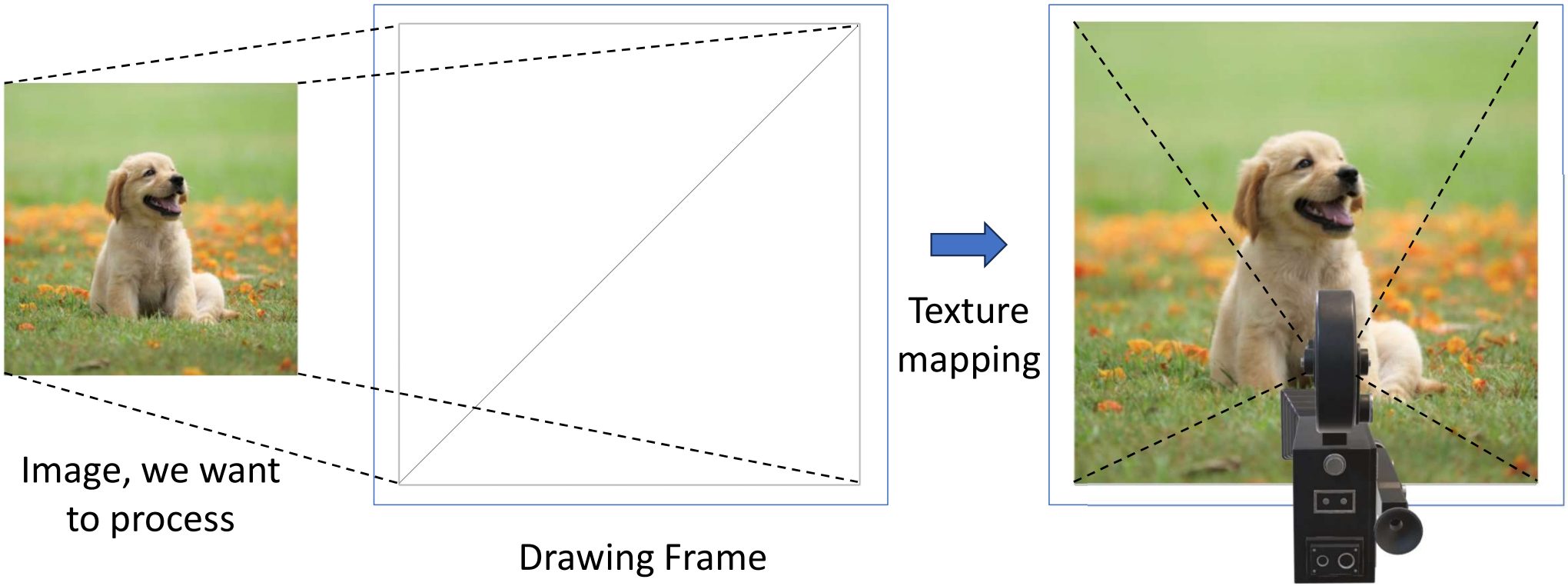


A quad (two triangles)

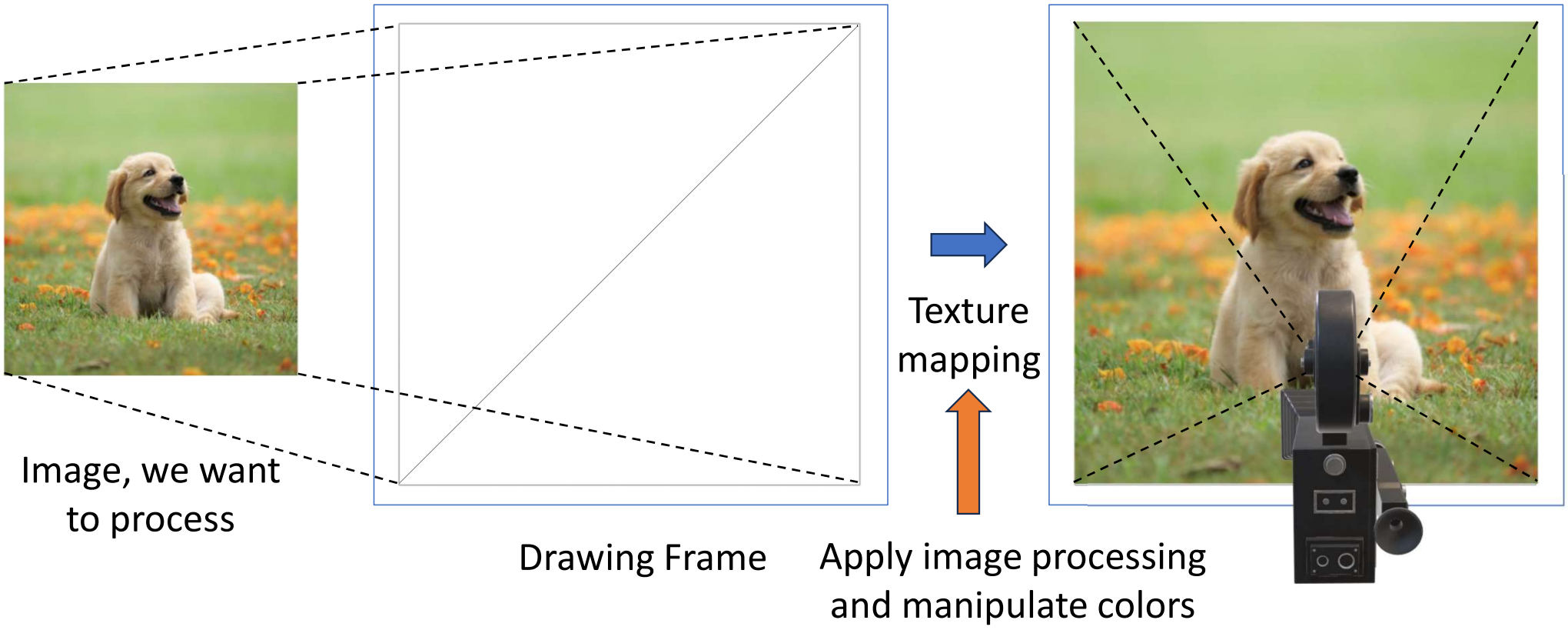


Shaded quad

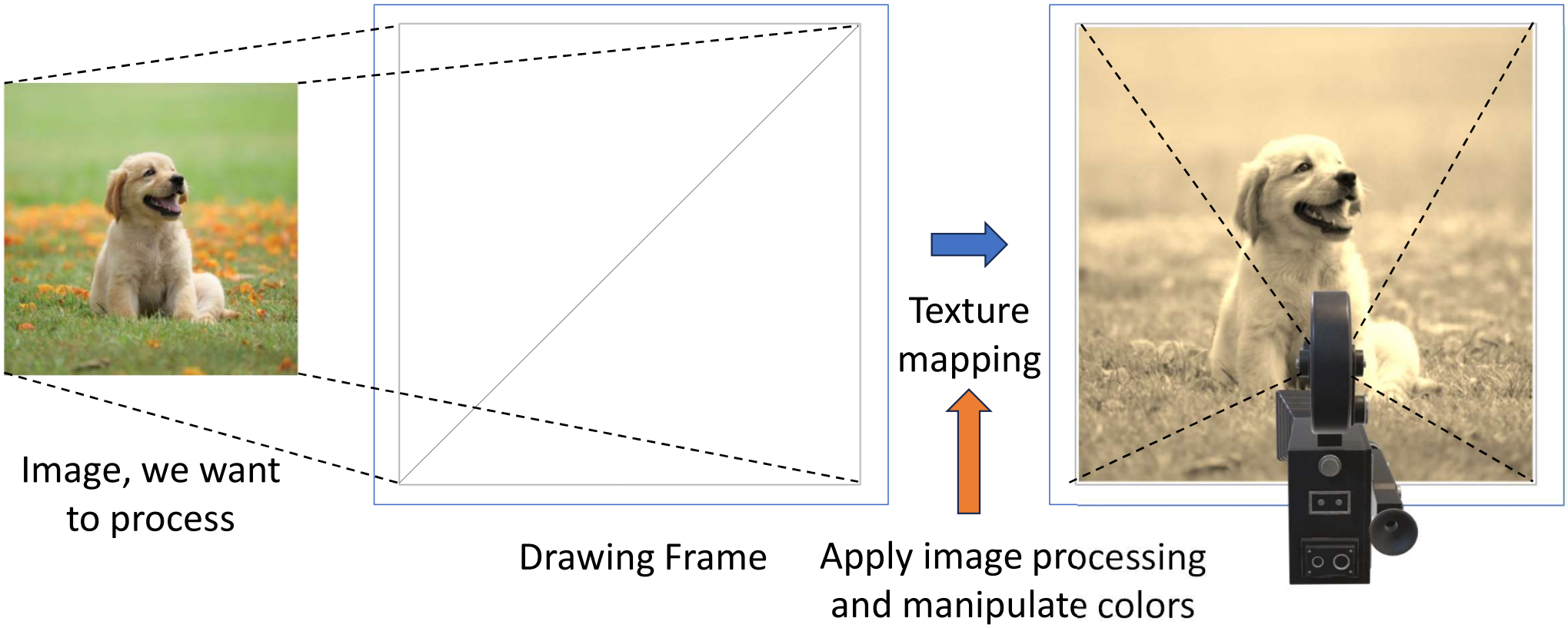
Post Processing of Fragments: Pipeline



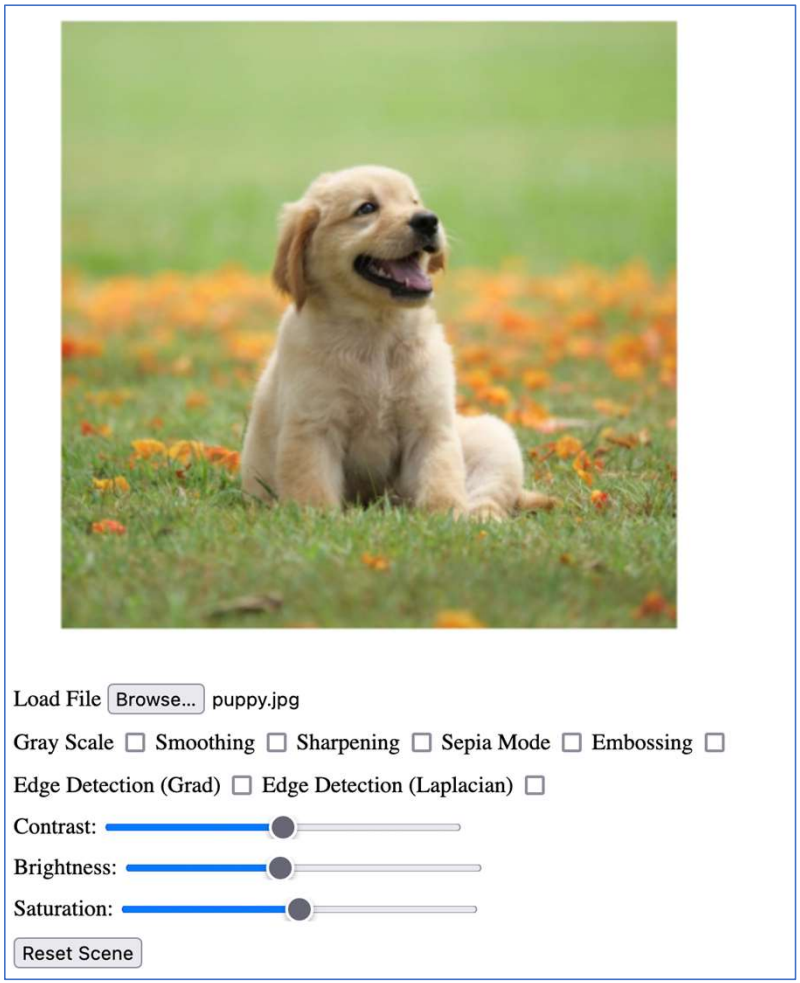
Post Processing of Fragments: Pipeline



Post Processing of Fragments: Pipeline



Post Processing Demo

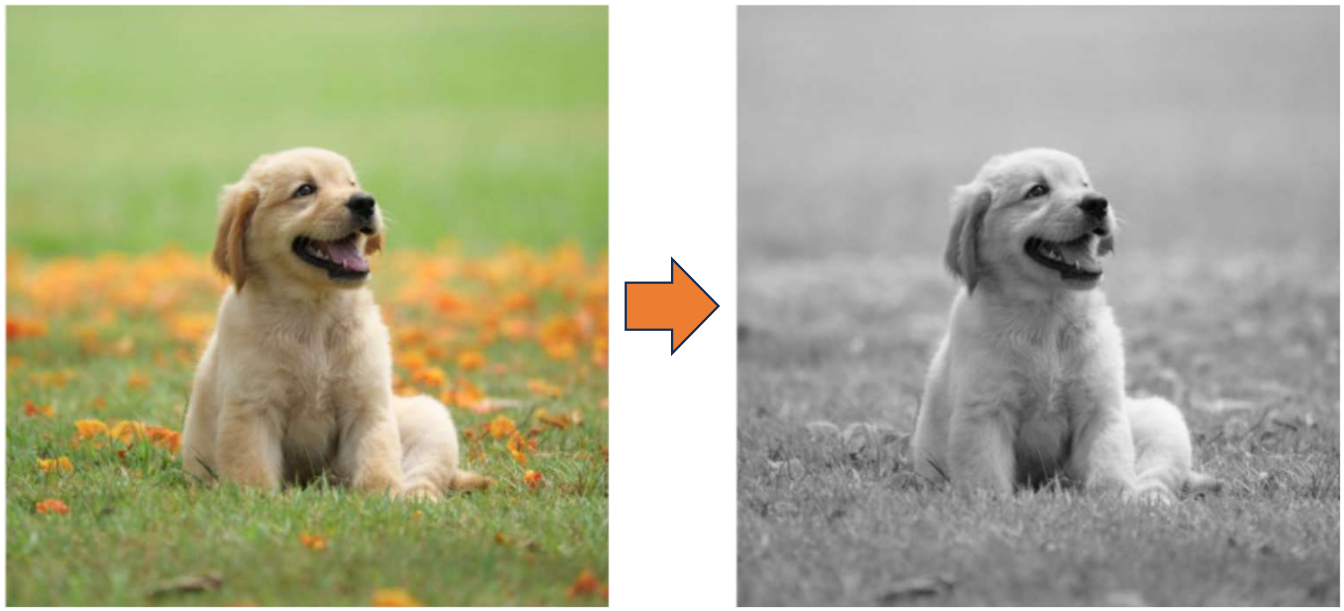


Some Operations on Images

- Pixel value modification
  - RGB to Gray Scale
  - Sepia Mode
  - Controlling Contrast, Brightness, Saturation, etc.
- Convolution-based operation
  - Smoothing
  - Sharpening
  - Embossing
  - Edge Detection

RGB to Gray Scale

- A very basic operation
- Multiply texture color with vec3(0.2126, 0.7152, 0.0722)

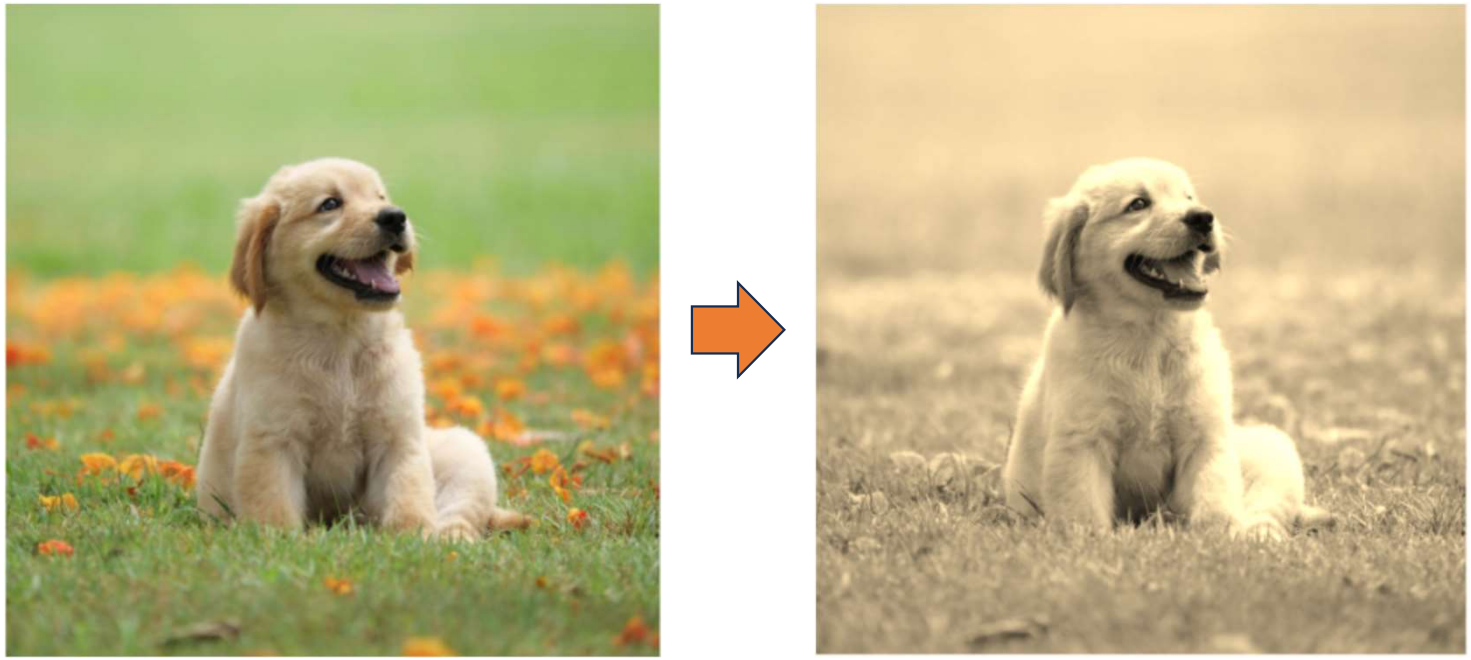


Sepia Filter

- Sepia is a reddish-brown or yellowish-brown color that is often associated with old-fashioned or vintage photographs
- It is typically achieved by manipulating the RGB pixel values in a specific way
- sepiaR = 0.393*texColor.r + 0.769*texColor.g + 0.189*texColor.b;
- sepiaG = 0.349*texColor.r + 0.686*texColor.g + 0.168*texColor.b;
- sepiaB = 0.272*texColor.r + 0.534*texColor.g + 0.131*texColor.b;

Sepia Filter

- Sepia is a reddish-brown or yellowish-brown color that is often associated with old-fashioned or vintage photographs



Contrast Adjustment

- Contrast is the degree of difference between the brighter and darker parts of an image
  - Increasing contrast will increase the difference
  - Decreasing contrast will decrease the difference
- 0.5 + (contrast + 1.0) * (texColor.rgb - 0.5)
  - Shift to [-0.5,0.5] range
  - Scale the values
  - Shift back to [0,1] range

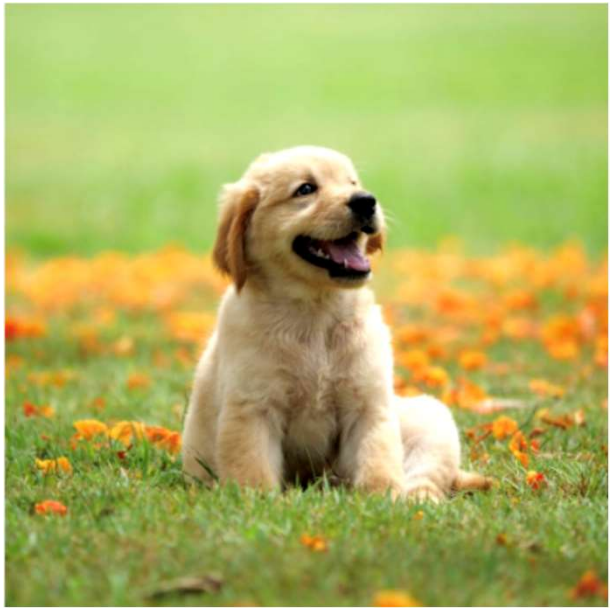
Contrast Adjustment

IITK CS360A

- Contrast is the degree of difference between the brighter and darker parts of an image

Contrast Reduced

Original

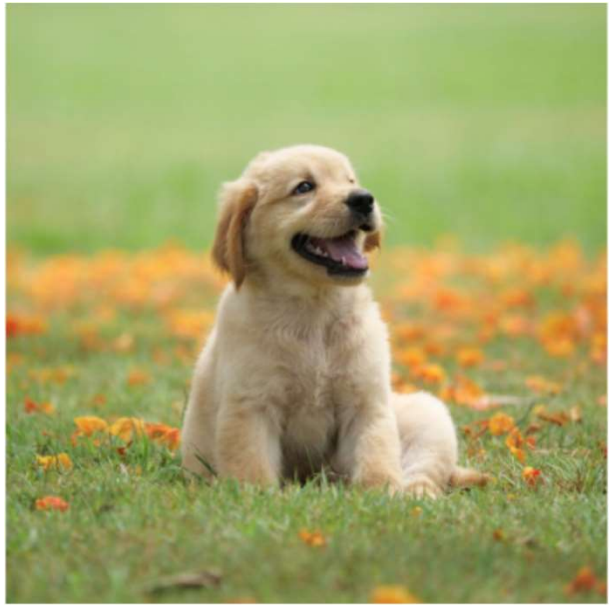


Contrast Enhanced

Brightness Adjustment

- Brightness refers to the overall ‘lightness’ or ‘darkness’ of the image
- Increasing/decreasing brightness can be achieved by simply increasing/decreasing the pixel values

Decreased brightness



Original

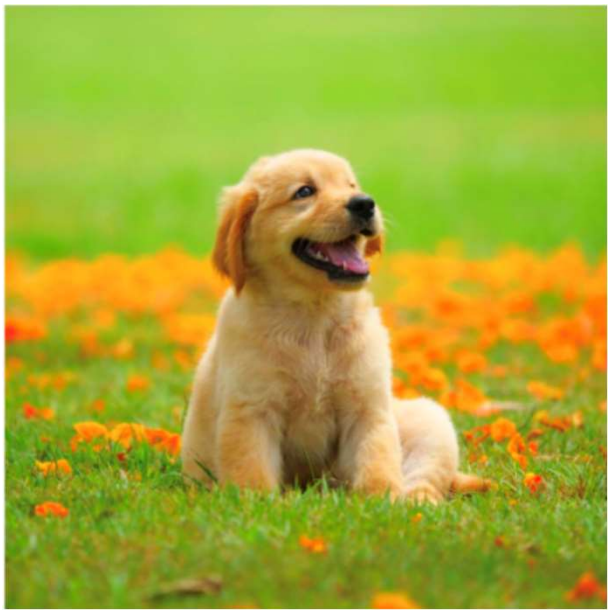
Increased brightness

Saturation Adjustment

- Saturation adjustment can either raise or reduce the intensity of the color values of the pixels
- mix(grayscaleColor, texColor, 1.0 + saturation)

Saturation=-1

Saturation=0



Saturation=1

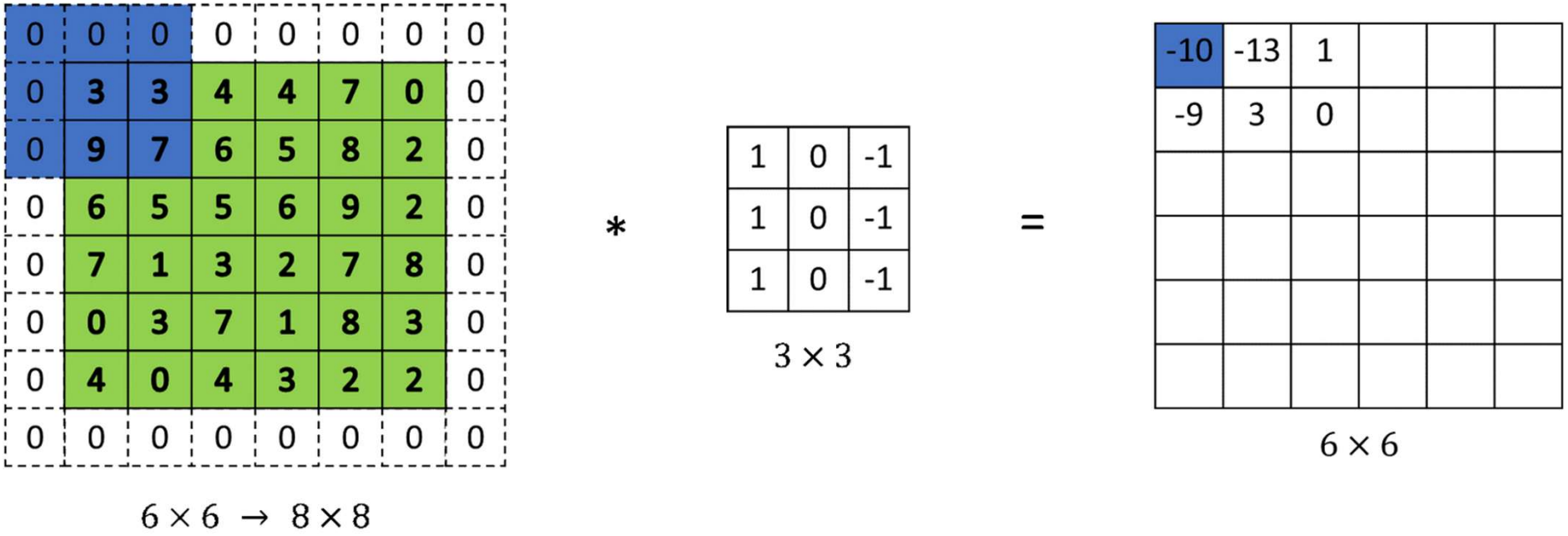
Image Convolution

- Image convolution is a fundamental operation in image processing and computer vision
- Apply a small matrix called a kernel or filter to an input image to produce a modified version of that image
- Convolution operation is used for various purposes, such as blurring, sharpening, edge detection, and feature extraction

Image Convolution

- The basic idea behind image convolution is to slide the kernel over the entire input image and perform element-wise multiplication between the kernel and the corresponding image region
- The results of these multiplications are then summed up to obtain a single value, which becomes the new pixel value at the center of the kernel's location in the output image

Image Convolution



Smoothing

- Each pixel color is replaced by the average of its neighborhood pixel values

*

1	1	1
1	1	1
1	1	1

Box Filter

=

Sharpening

- A special sharpening kernel is used for convolution

$*$

0	-1	0
-1	5	-1
0	-1	0

$=$

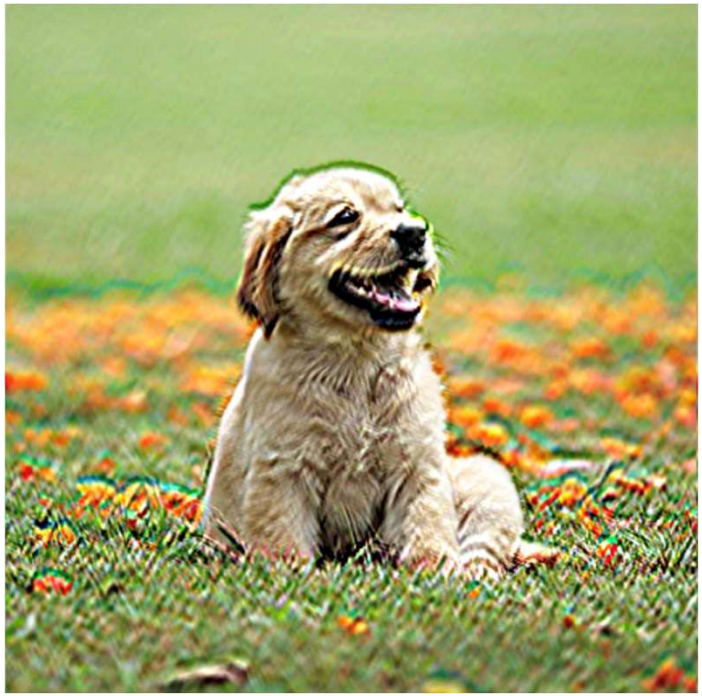
Embossing

- A special embossing kernel is used for convolution

*

-3	-1	0
-1	5	1
0	1	3

=



Edge Detection using Gradient

- Compute gradient magnitude using central differencing technique

```
vec4 dy = (up-down)*0.5;
vec4 dx = (right-left)*0.5;
vec4 gradientMag = sqrt(dx*dx+
dy*dy);
```

Edge Detection using Laplacian

- Use a special kernel

*

0	-1	0
-1	4	-1
0	-1	0

=