CPSC 4040/6040 Computer Graphics Images

Joshua Levine
levinej@clemson.edu

Lecture 13
Sampling and Interpolation

Oct. 1, 2015

Agenda

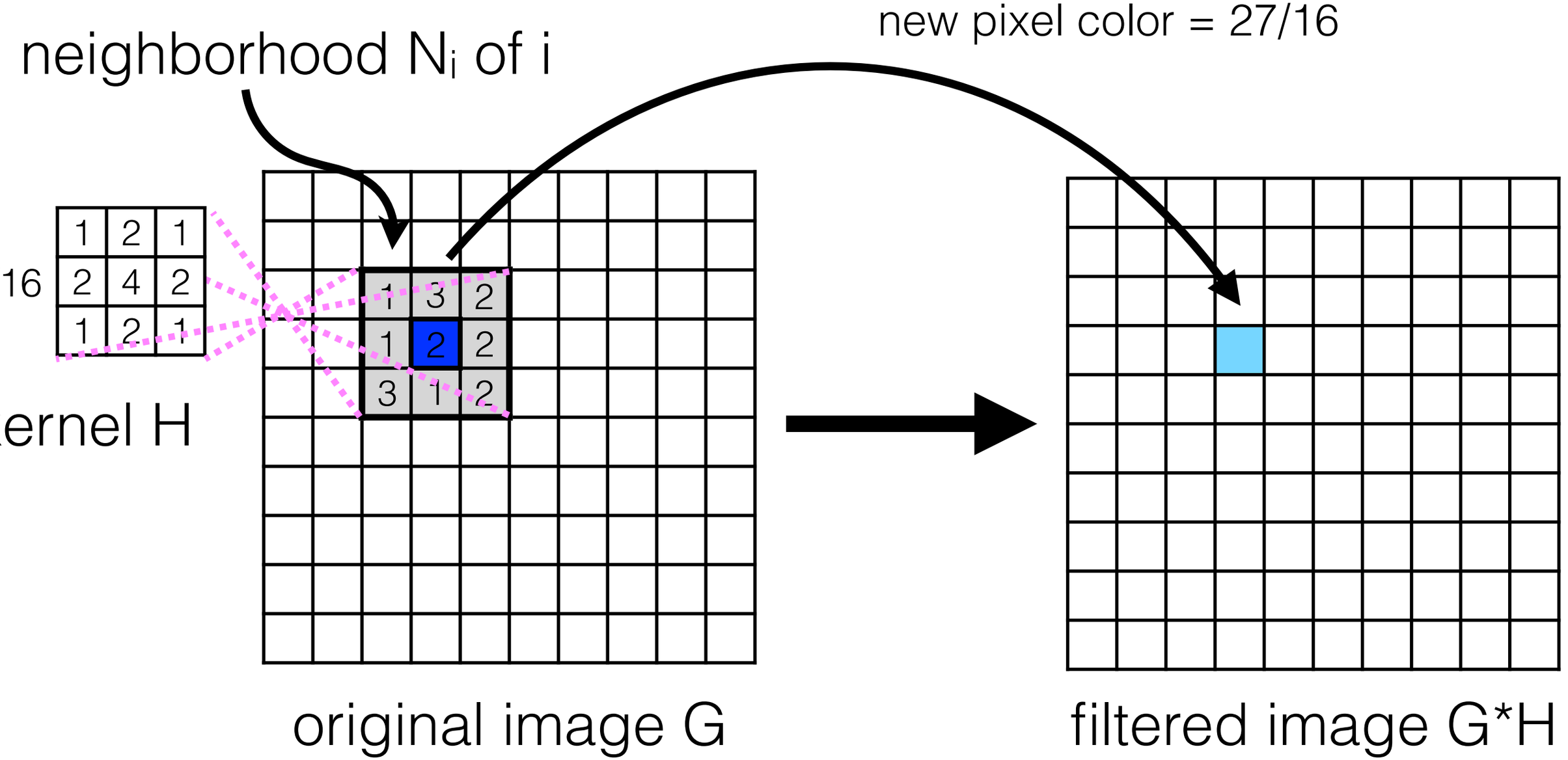
- Q02 Due

- PA04 questions?

Last Time

Convolution

- This process of adding up pixels multiplied by various weights is called **convolution**



Smoothing Spatial Filters

- Any weighted filter with positive values will smooth in some way, examples:

$$\frac{1}{9}\times\begin{array}{|c|c|c|}\hline 1 & 1 & 1\\\hline 1 & 1 & 1\\\hline 1 & 1 & 1\\\hline\end{array}$$

$$\frac{1}{16}\times\begin{array}{|c|c|c|}\hline 1 & 2 & 1\\\hline 2 & 4 & 2\\\hline 1 & 2 & 1\\\hline\end{array}$$

- Normally, we use integers in the filter, and then divide by the sum (computationally more efficient)

- These are also called **blurring** or **low-pass** filters

Smoothing Example

(a) Source image.

(b) 17 × 17 Box.

(c) 17 × 17 Gaussian.

Figure 6.10. Smoothing examples.

Sharpening Filters

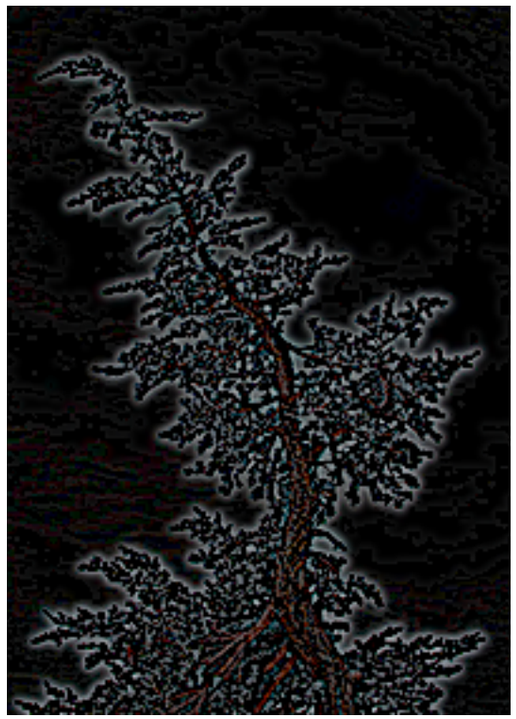
Sharpening (Idea)

Input

-

blurred

=

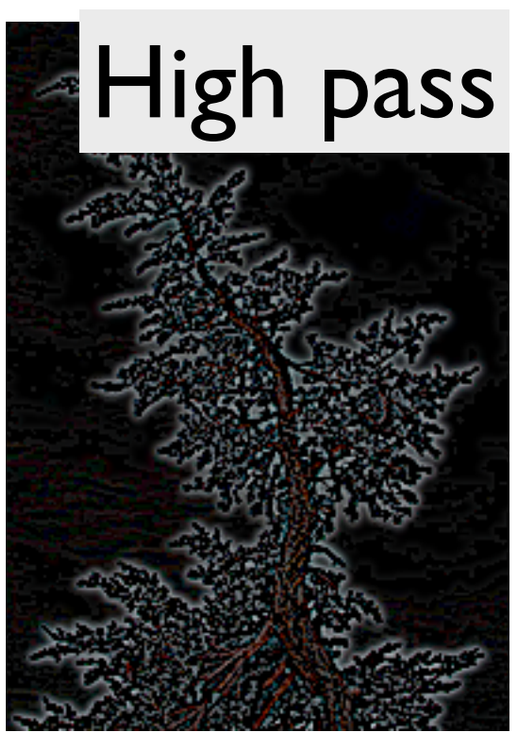


High pass

Input

+k*

High pass



=

Sharpened image

Sharpening is a Convolution

- This equation can then be used to generate appropriate sharpening kernels
  - Assume that $I = I \otimes K_{identity}$ and $I_{low} = I \otimes K_{low}$.
  - Note that $I_{high} = I - I_{low}$. $I_{sharp} = I + \boldsymbol{\alpha}^* I_{high}$. This leads to:

$$
\begin{aligned}
I_{\text{sharp}} &= (1+\alpha)I - \alpha I_{\text{low}}, \\
&= (1+\alpha)(I \otimes K_{\text{identity}}) - \alpha(I \otimes K_{\text{low}}), \\
&= I \otimes (1+\alpha)K_{\text{identity}} - I \otimes (\alpha K_{\text{low}}), \\
&= I \otimes ((1+\alpha)K_{\text{identity}} - \alpha K_{\text{low}}).
\end{aligned}
$$

Sharpening is a Convolution

$$
\begin{aligned}
I_{\text{sharp}} &= (1+\alpha)I - \alpha I_{\text{low}}, \\
&= (1+\alpha)(I \otimes K_{\text{identity}}) - \alpha(I \otimes K_{\text{low}}), \\
&= I \otimes (1+\alpha)K_{\text{identity}} - I \otimes (\alpha K_{\text{low}}), \\
&= I \otimes ((1+\alpha)K_{\text{identity}} - \alpha K_{\text{low}}).
\end{aligned}
$$

Note: could also define K_{identity} as $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

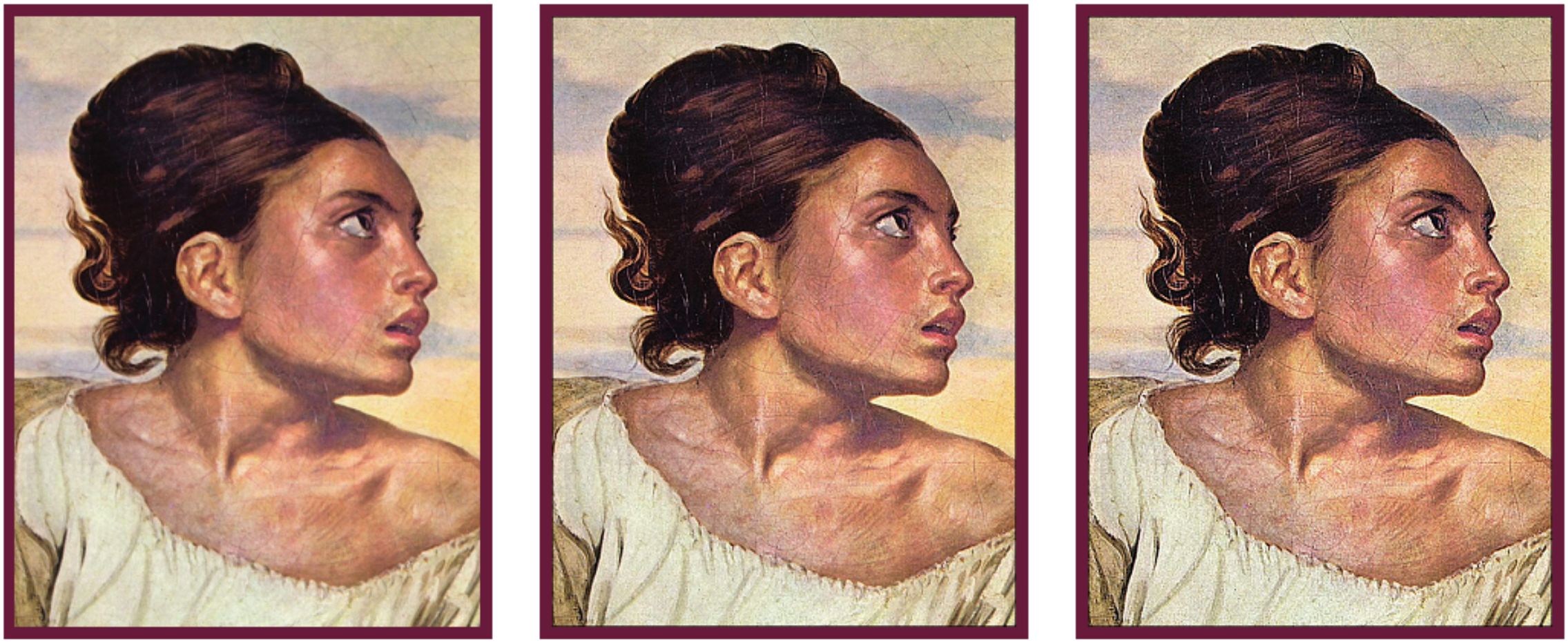
$$
K_{\text{identity}} = \frac{1}{9} \times \begin{bmatrix} 0 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 0 \end{bmatrix},
$$

$$
K_{\text{low}} = \frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix},
$$

$$
(1+\alpha)K_{\text{identity}} - \alpha K_{\text{low}} = \frac{1}{9} \times \begin{bmatrix} -\alpha & -\alpha & -\alpha \\ -\alpha & (9+8\alpha) & -\alpha \\ -\alpha & -\alpha & -\alpha \end{bmatrix}
$$

Edge Enhancement

- The parameter α can be considered a gain setting that controls how much of the source image is passed through to the sharpened image.

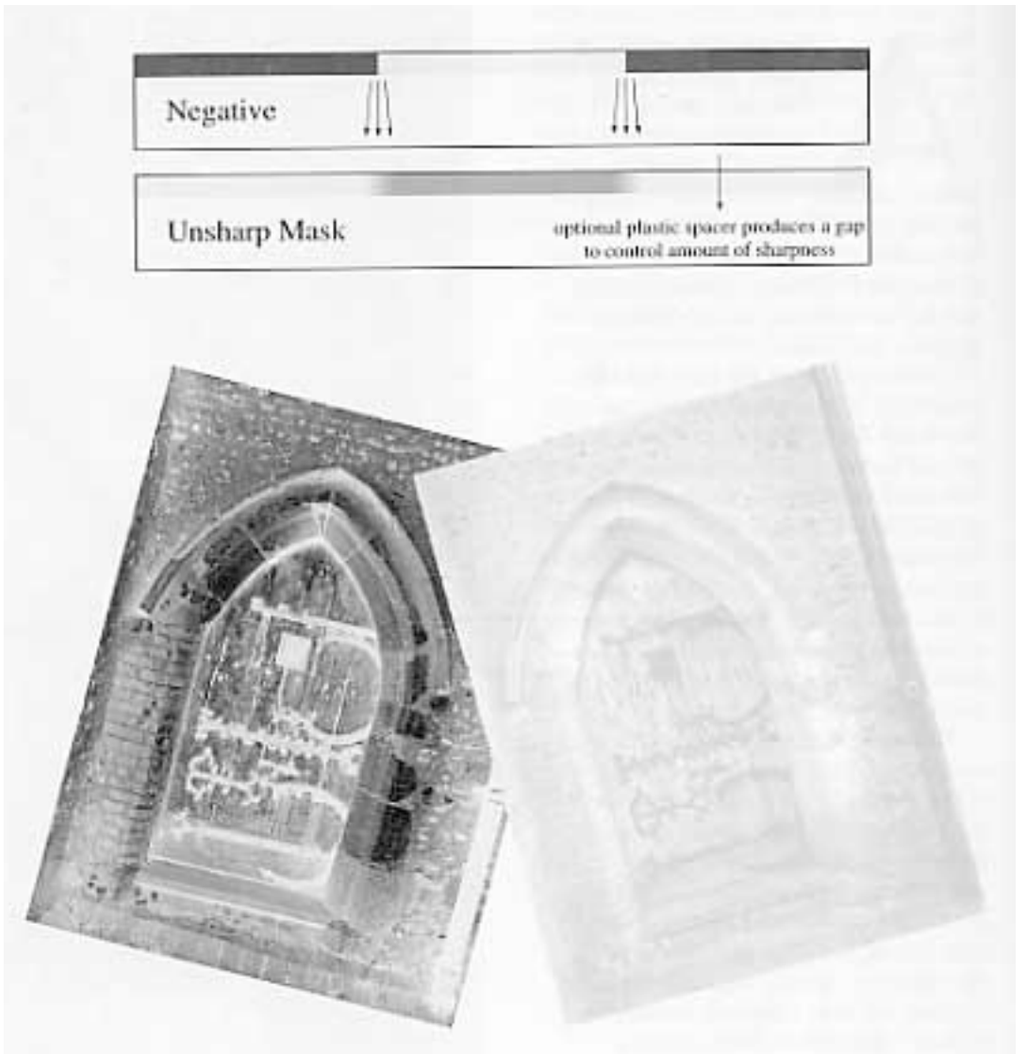


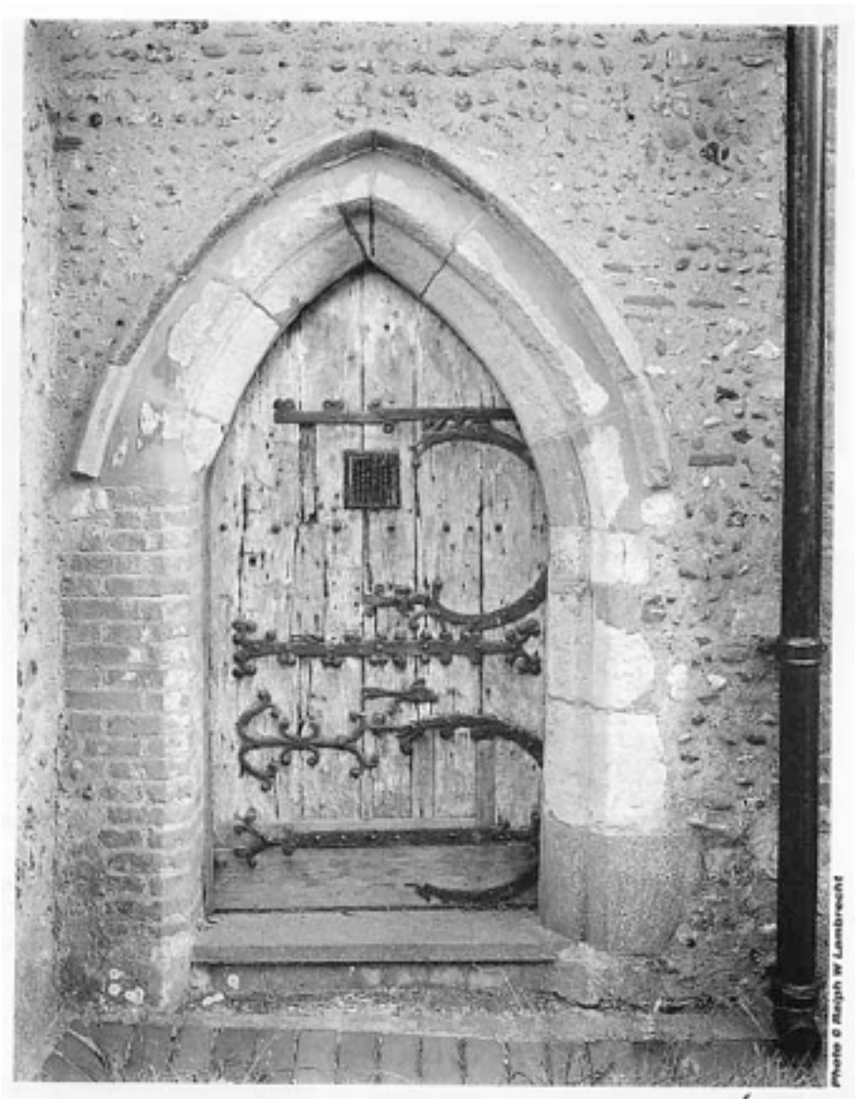
(a) Source image. (b) $\alpha = .5$. (c) $\alpha = 2.0$.

Figure 6.20. Image sharpening.

Unsharp Masks

- Sharpening is often called “unsharp mask” because photographers used to sandwich a negative with a blurry positive film in order to sharpen





http://www.tech-diy.com/UnsharpMasks.htm

Fig.4: The two examples here show a detail of the brick-work to the left of the church door. The one on the left was printed with the negative alone – the one on the right was printed with both negative and mask as a sandwich. The increase in local contrast and edge sharpness is minute, but clearly visible. Grade 2.5 was used for the straight print but increased to 4.5 for the sandwiched image to compen-sate for the reduced contrast.

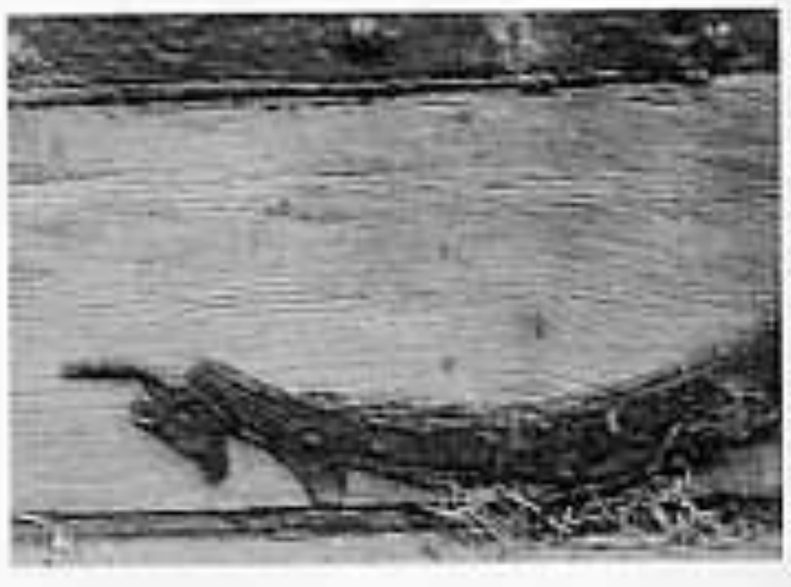


Fig.5: These two examples show a detail of the lower right hand side of the church door. Here the difference in sharpness is clearly visible between the (left) negative and (right) sandwich prints.

Defining Edges

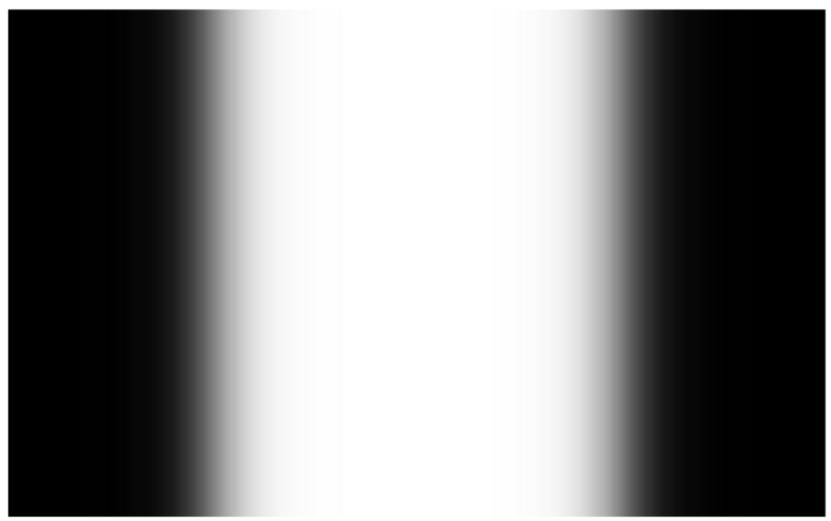
- Sharpening uses negative weights to enhance regions where the image is changing rapidly
  - These rapid transitions between light and dark regions correspond to **edges**

- Smoothing reduces the strength of edges, sharpening strengthens them.

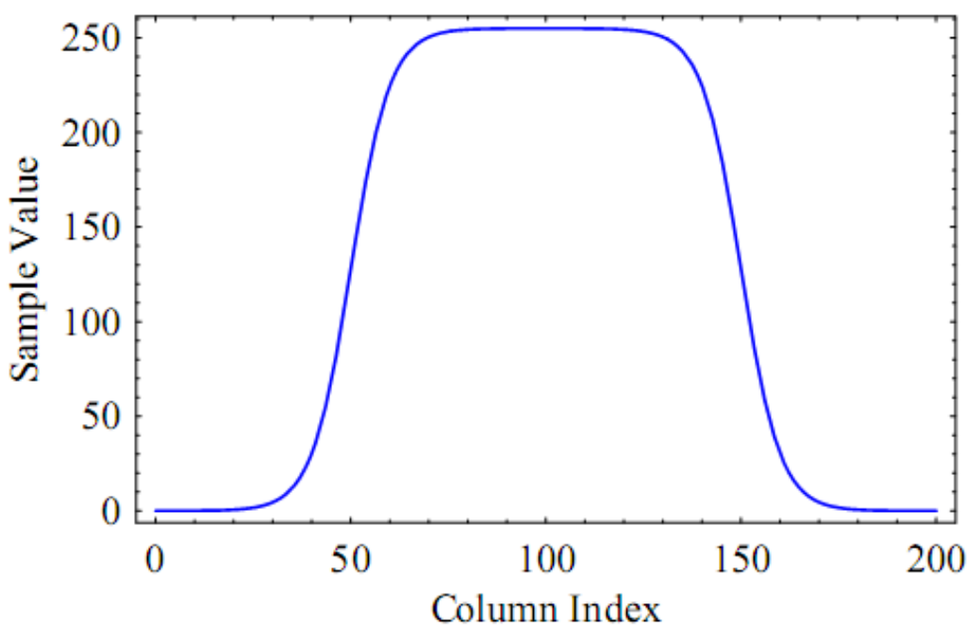
- Also called **high-pass** filters

- Idea: smoothing filters are weighted averages, or integrals. Sharpening filters are weighted differences, or derivatives!

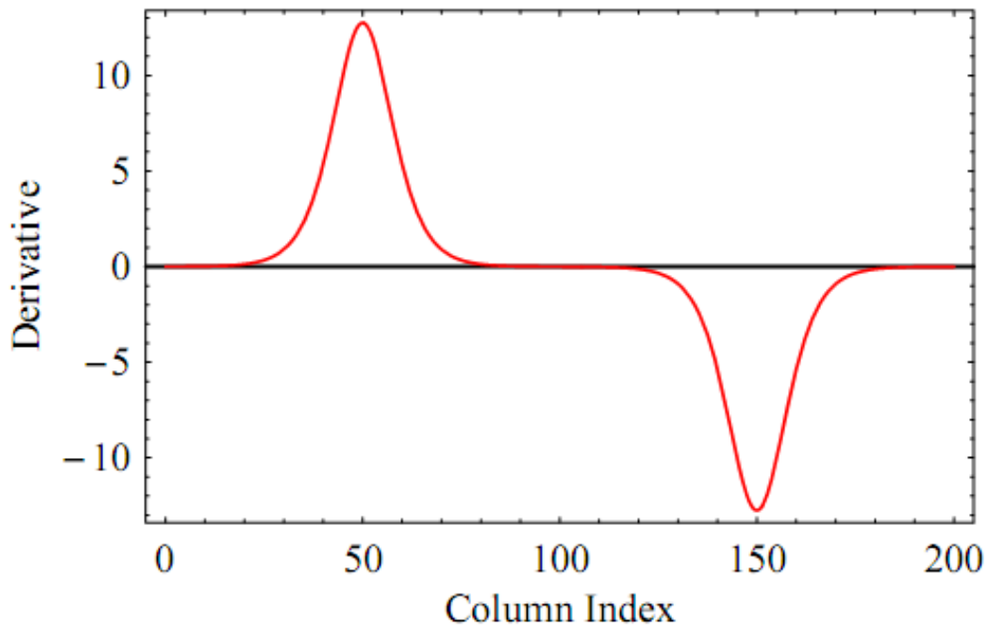
Edges



(a)



(b)



(c)

Figure 6.11. (a) A grayscale image with two edges, (b) row profile, and (c) first derivative.

(Review?) Derivatives via Finite Differences

- We can convert the derivative to a kernel w:

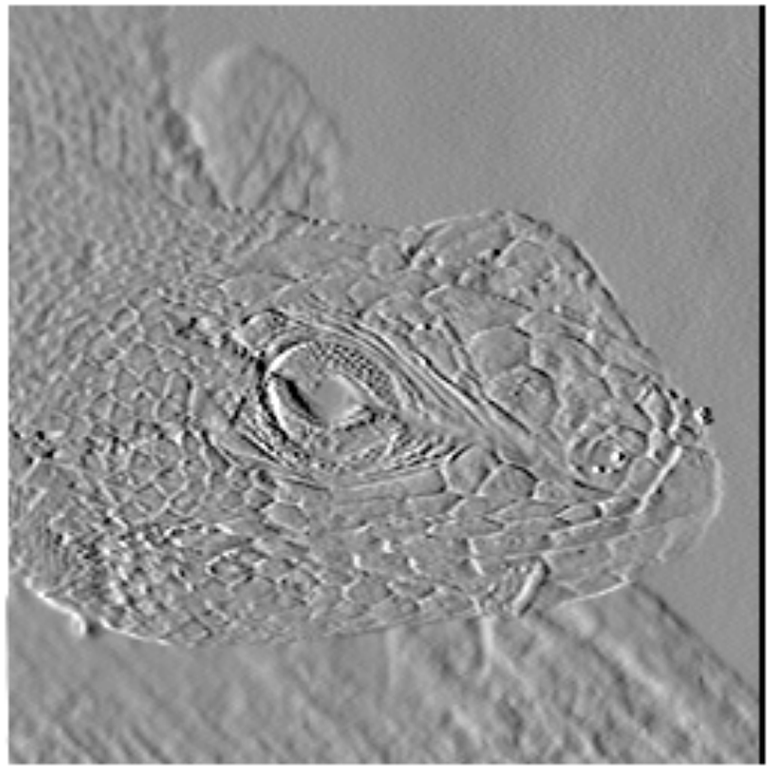
$$\frac{\partial f}{\partial x} \approx \frac{1}{2h}\left(f(x+1,y) - f(x-1,y)\right)$$

$$\frac{\partial f}{\partial x} \approx w_{dx} \circ f \qquad w_{dx} = \begin{array}{|c|c|c|} \hline -\frac{1}{2} & 0 & \frac{1}{2} \\ \hline \end{array}$$

$$\frac{\partial f}{\partial y} \approx w_{dy} \circ f \qquad w_{dy} = \begin{array}{|c|} \hline -\frac{1}{2} \\ \hline 0 \\ \hline \frac{1}{2} \\ \hline \end{array}$$

Taking Derivatives with Convolution

$$\begin{matrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{matrix}$$



$$\begin{matrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{matrix}$$

Gradients with Finite Differences

- These partial derivatives approximate the image gradient, ∇I.
- Gradients are the unique direction where the image is changing the most rapidly, like a slope in high dimensions
- We can separate them into components kernels G_x, G_y. $\nabla I = (G_x, G_y)$

$$\nabla I(x,y) = \begin{pmatrix} \delta I(x,y)/\delta x \\ \delta I(x,y)/\delta y \end{pmatrix}.$$

$$G_x = [1, 0, -1] \qquad G_y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix};$$

$$\nabla I = \begin{pmatrix} \delta I/\delta x \\ \delta I/\delta y \end{pmatrix} \simeq \begin{pmatrix} I \otimes G_x \\ I \otimes G_y \end{pmatrix}.$$

Figure 6.12. Image gradient (partial).

128	187	210	238	251
76	121	193	225	219
66	91	110	165	205
47	81	83	119	157
41	59	63	75	125

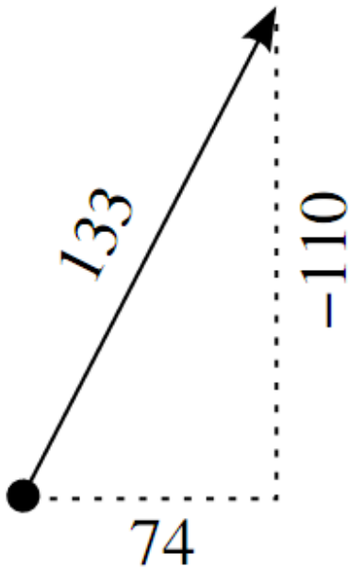
(a) Source Image.

	117	104	26	
	44	74	95	
	36	38	74	

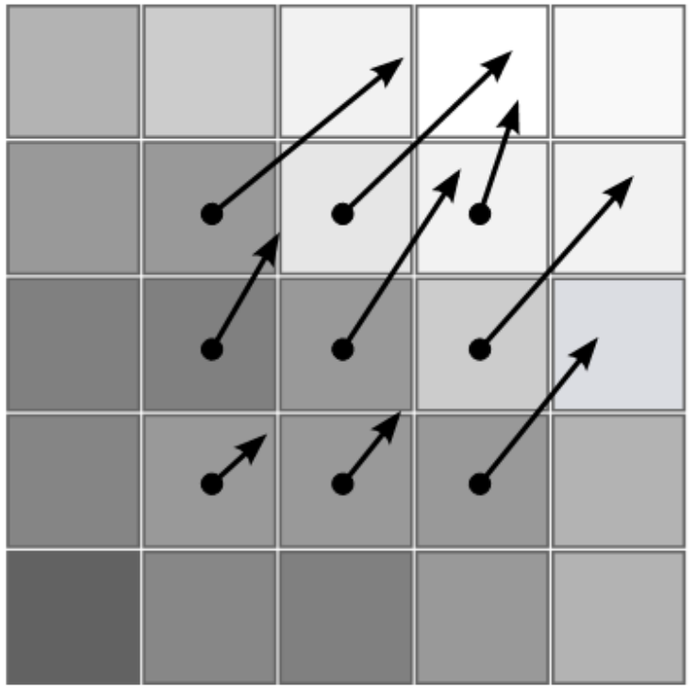
(b) $\delta I / \delta x$.

	−96	−100	−73	
	−40	−110	−106	
	−32	−47	−90	

(c) $\delta I / \delta y$.



(d) Center sample gradient.



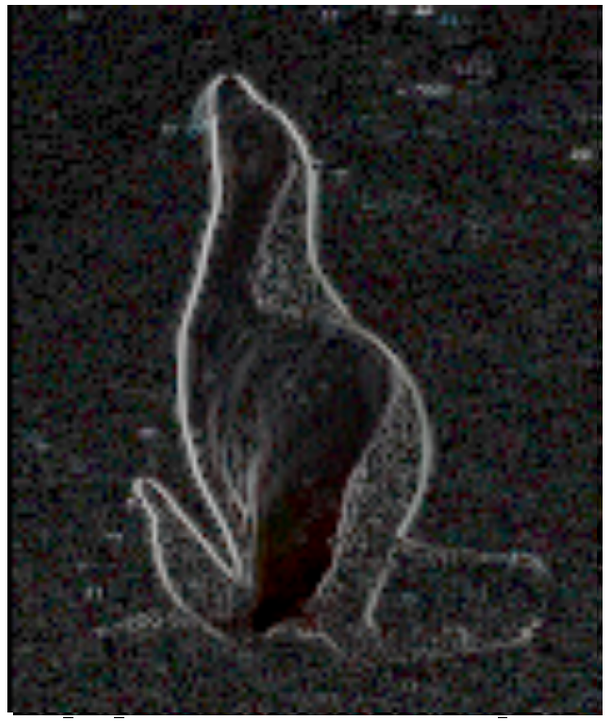
(e) Gradient.

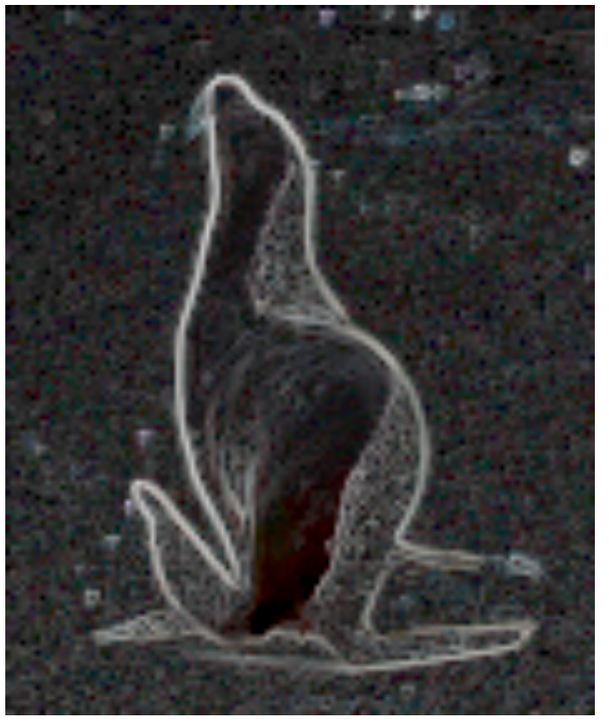
151	144	77
59	133	142
48	60	117

(f) Magnitude of gradient.

Figure 6.14. Numeric example of an image gradient.

Gradients Gx, Gy





$|G_x|$ $|G_y|$ $|G| = \sqrt{(G_x^2 + G_y^2)}$

Prewitt Operator

- The **Prewitt** operators are another way to compute derivatives.
  - They average the difference across the central element and hence provide some tolerance to noise.

$$G_x = \frac{1}{3}\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}, \qquad G_y = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

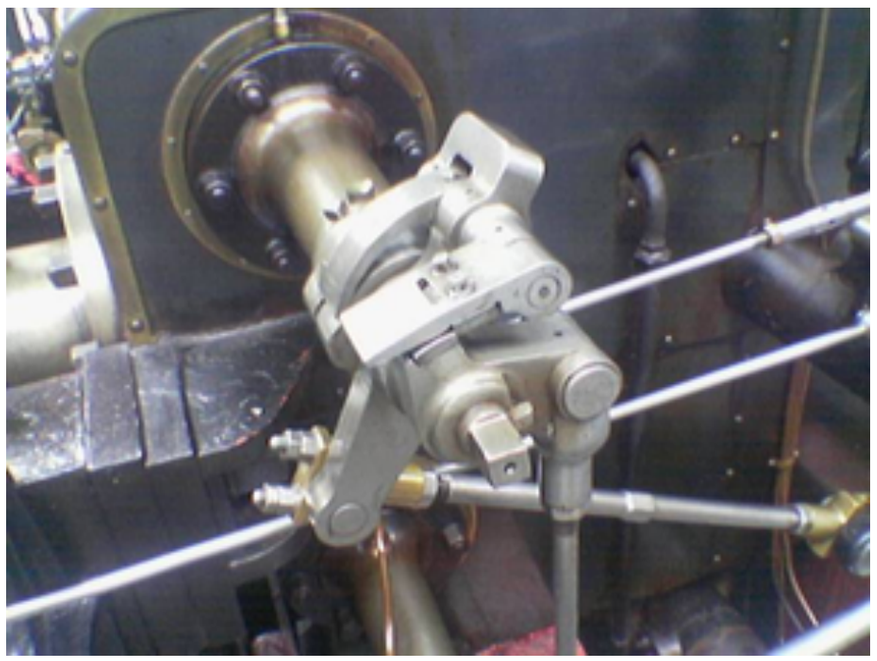
(a) Source image I. (b) $I \otimes G_x$. (c) $I \otimes G_y$.

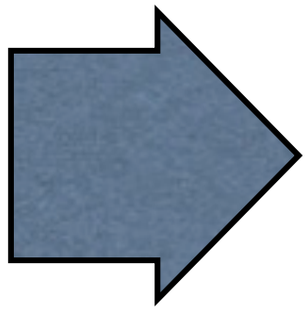
Figure 6.15. Prewitt operators.

Sobel Operator

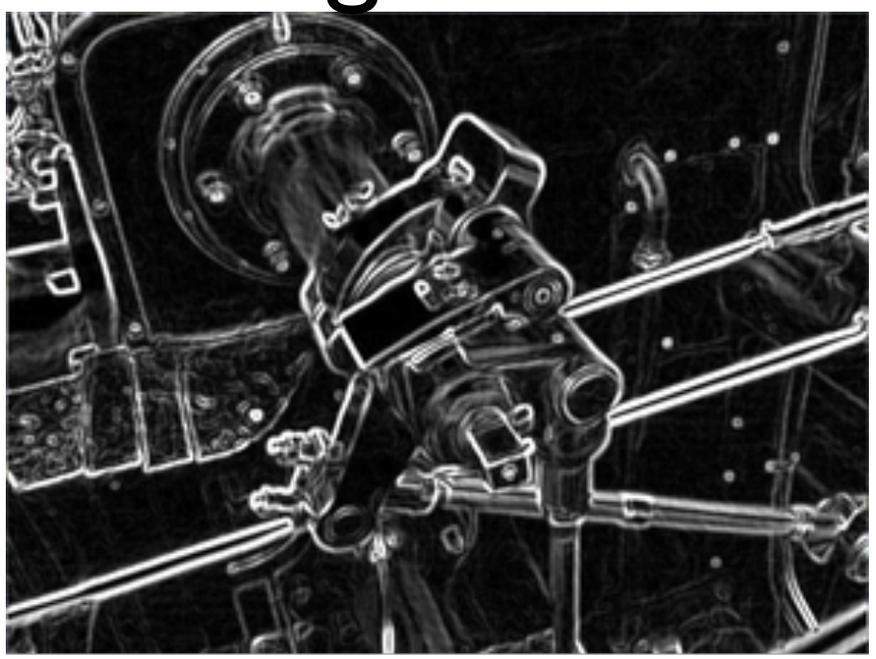
- Weighted to emphasize the center element

$$G_x = \frac{1}{4}\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}, \quad G_y = \frac{1}{4}\begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$





Magnitude



Roberts Cross-Gradient Operators

- Idea: measure change in the upper-left-to-bottom-right and bottom-left-to-upper-right diagonals rather than the horizontal and vertical axes.
  - Are 2x2 with the center element in the upper-left

$$G_1 = \begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix}, \quad G_2 = \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix}$$

(a) Source image I. (b) $I \otimes G_1$. (c) $I \otimes G_2$.

Figure 6.16. A source image is convolved with the Roberts operators G_1 and G_2.

Second Derivatives (Sharpening, almost)

- Partial derivatives in x and y lead to two kernels:

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

and, similarly, in the y-direction we have

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

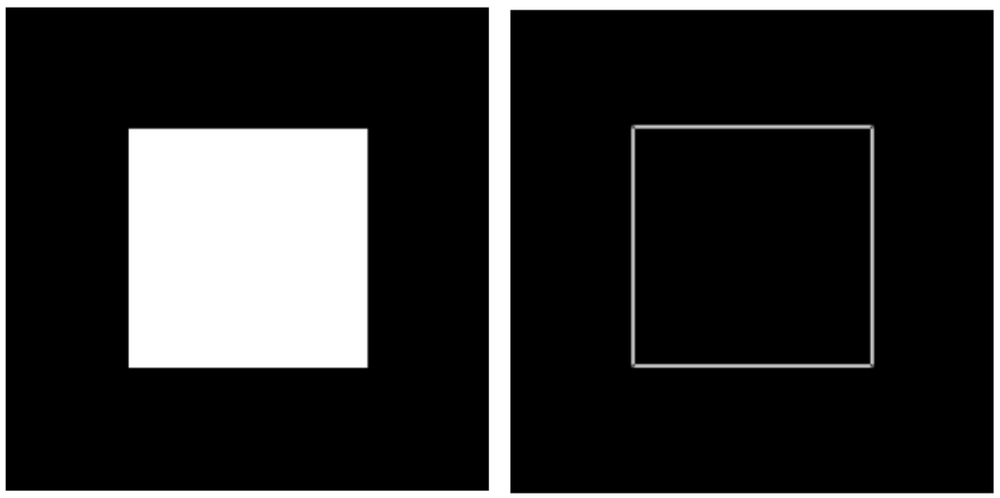
Compare with Sharpening filter: unbalanced counts!

$$\begin{bmatrix} -\alpha & -\alpha & -\alpha \\ -\alpha & (9+8\alpha) & -\alpha \\ -\alpha & -\alpha & -\alpha \end{bmatrix}$$

1	1	1
1	-9	1
1	1	1

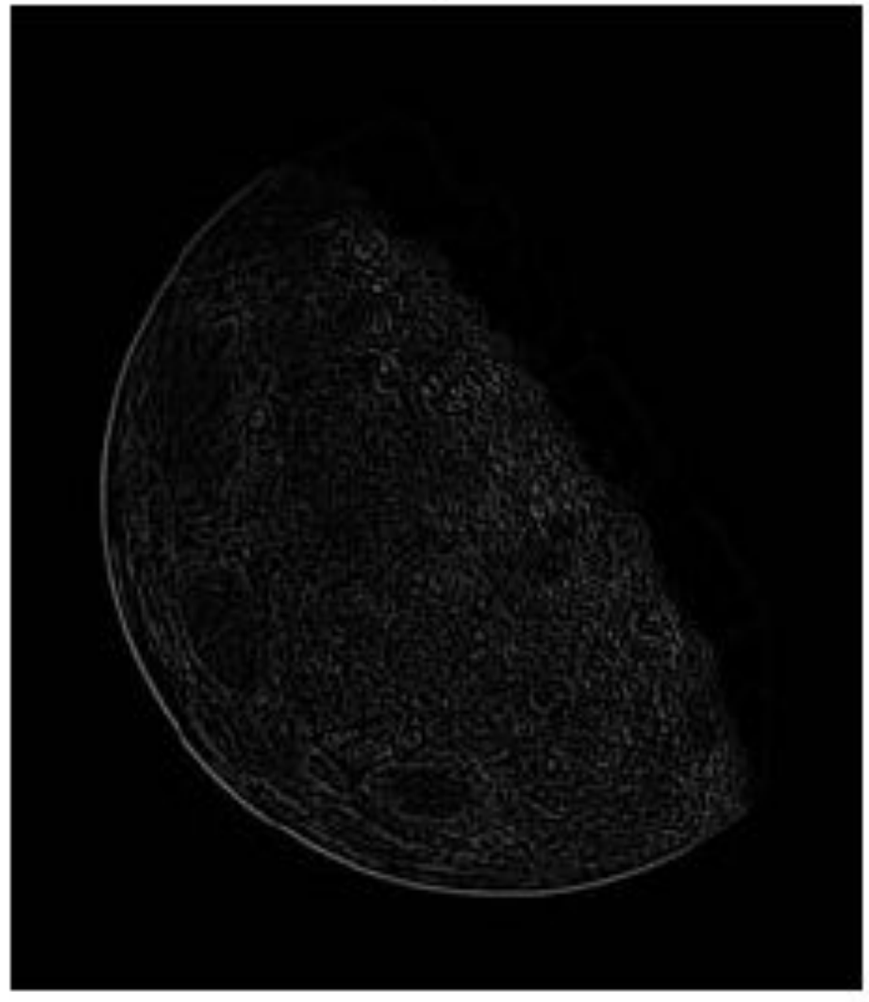
Examples

a)



b)

0	1	0
1	−4	1
0	1	0



Edge Detection

- Edge detection algorithms typically compute the magnitude of the gradient (MoG) since larger magnitudes directly correspond to greater probability of an edge.

- An edge map is the inverse of the MoG

- Can detect edges by thresholding the MoG. Magnitudes above the threshold are assumed to be on an edge.

(a) Source image.

(b) Magnitude of gradient.

(c) Edge map.

(d) Thresholded MOG.

(a) Roberts MoG.

(c) Roberts G_1.

(e) Roberts G_2.

(b) Prewitt MoG.

(d) Prewitt G_x.

(f) Prewitt G_y.

Figure 6.18. Edge maps using Prewitt and Roberts kernels.

Using Edges for Effects

(a) Source image.

(b) Texture.

(b) Glow.

Figure 6.19. Artistic effects using edges.

Kernel Separability

Convolution Complexity

- What is the computational complexity of the “brute-force” approach to convolving a WxH image with MxN kernel?

Convolution Complexity

- What is the computational complexity of the “brute-force” approach to convolving a WxH image with MxN kernel?
    - Computing each destination sample requires M*N multiplications and additions

Convolution Complexity

- What is the computational complexity of the “brute-force” approach to convolving a WxH image with MxN kernel?
    - Computing each destination sample requires M*N multiplications and additions
    - There are W*H samples.

Convolution Complexity

- What is the computational complexity of the “brute-force” approach to convolving a WxH image with MxN kernel?
    - Computing each destination sample requires M*N multiplications and additions
    - There are W*H samples.
    - Convolution is on the order of W*H*M*N operations.

Convolution Complexity

- What is the computational complexity of the “brute-force” approach to convolving a WxH image with MxN kernel?
    - Computing each destination sample requires M*N multiplications and additions
        - There are W*H samples.
        - Convolution is on the order of W*H*M*N operations.
- This is SLOW!
    - Large kernels are impractical

Example: Separating Median Filters

51	55	59	59	69
48	53	60	61	72
43	52	253	65	70
39	50	55	59	68
35	49	58	48	57

(a) Source image.

•	55	59	59	•
•	53	60	61	•
•	52	65	70	•
•	50	55	59	•
•	49	49	57	•

(b) Row approximation.

•	•	•	•	•
•	53	60	61	•
•	52	60	61	•
•	50	55	59	•
•	•	•	•	•

(c) Final approximation.

Figure 6.25. Numeric example of fast (approximate) median filtering.

True Median Filter:

•	•	•	•	•
•	53	59	65	•
•	52	59	65	•
•	50	55	59	•
•	•	•	•	•

Convolution Can Often Be Perfectly Separated

- Computational efficiency can be dramatically improved if the kernel is separable.
- A MxN kernel is separable iff there exists an Mx1 column vector A and a 1xN row vector B such that K = AB.
- Example:

$$AB = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \times \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

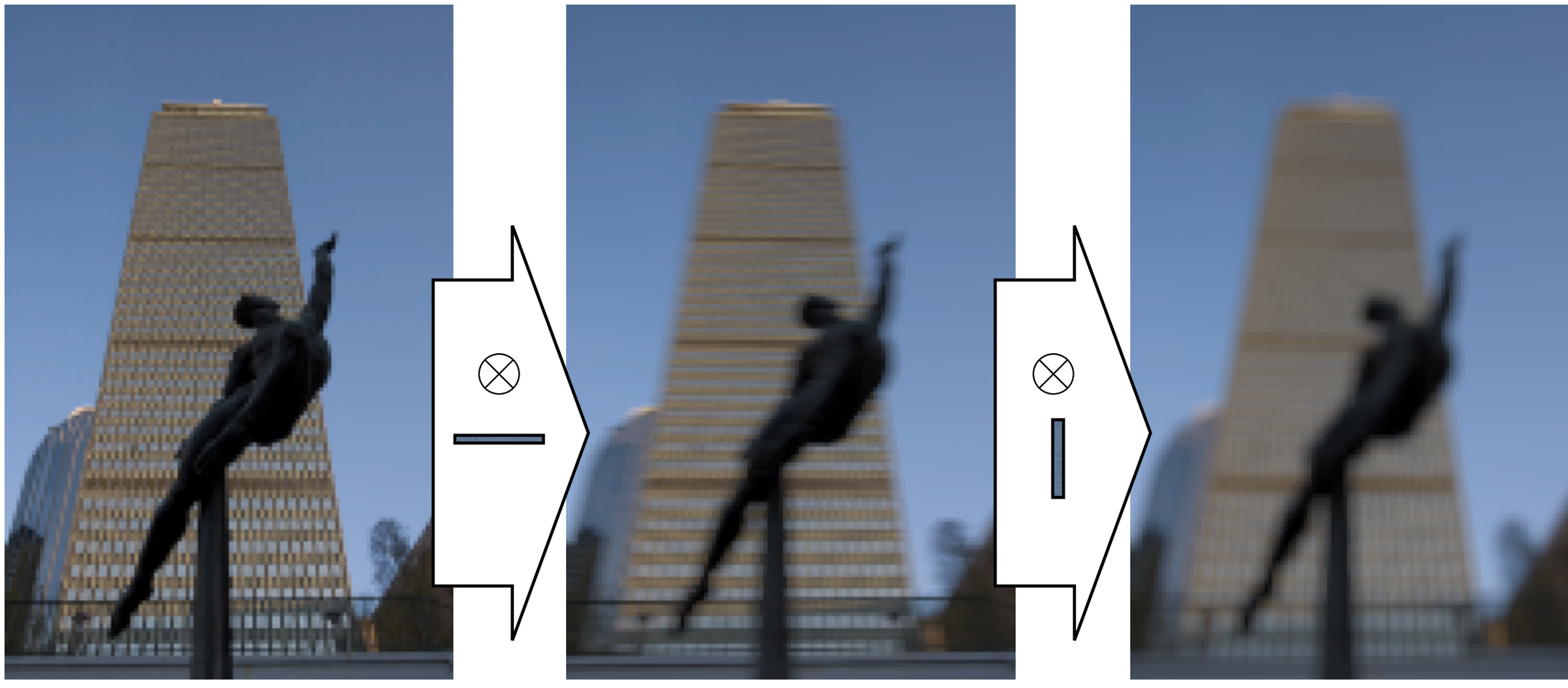
- The benefit derives from realizing that convolution can be performed by consecutive application of convolution of I with A followed by B.

$$I \otimes K = (I \otimes A) \otimes B \quad \text{where} \quad K = AB.$$

- This leads to M+N operations per pixel, or W*H*(M+N) total.

Separable Smoothing

- **First blur horizontally using g(x)**
- **Then blur vertically using g(y)**



Works for Gaussians too!

Sampling and Interpolation

Jim Blinn's Corner

http://www.research.microsoft.com/~blinn/

What Is a Pixel?

James F. Blinn

Microsoft Research

I am not a major sports fan. But there is one story from the folklore of football that has always intrigued me. The story goes that legendary football coach Vince Lombardi was observing his new players, recruited from the best college teams and all presumably excellent players. He was, however, not pleased with their performance. So he called them all to a meeting, which he began by holding up the essential tool of their trade and saying, "This is a football." I was sufficiently impressed by this back-to-basics attitude that, when I taught computer graphics rendering classes, I used to start the first lecture of the term by going to the blackboard (boards were black then, not white) and drawing a little dot and saying, "This is a pixel."

But was I right? Most computer graphicists would agree that the pixel is the fundamental atomic element of imaging. But what is a pixel really? As I have played with various aspects of pixel mashing, it has occurred to me that the concept of the pixel is really multifaceted (to use a weird metaphor.) Perhaps a better metaphor that spirit I am going to list some possible meanings for a pixel that I will expand on in some later columns.

A pixel is a little square

Early 2D windowing systems considered a pixel a little square. This is perhaps the simplest possible definition, but it only works if you are mostly drawing horizontal and vertical lines and rectangles that are integral numbers of pixels in size. Anything at fractional pixel size or at an angle yields jaggies and other forms of aliasing.

A pixel is a point sample of a continuous function

A more enlightened signal processing approach thinks of a pixel as a point sample of a continuous function (see the dots in Figure 1). (This approach was actually taken by the rendering community long before pixel displays were used for user interfaces and windowing systems.) Applying linear filtering theory to pixel manip-

Lec14 Required Reading

High Dynamic Range Image Encodings

Greg Ward, Anyhere Software

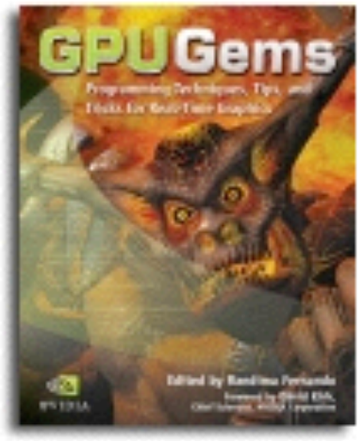
Introduction

We stand on the threshold of a new era in digital imaging, when image files will encode the color gamut and dynamic range of the original scene, rather than the limited subspace that can be conveniently displayed with 20 year-old monitor technology. In order to accomplish this goal, we need to agree upon a standard encoding for high dynamic range (HDR) image information. Paralleling conventional image formats, there are many HDR standards to choose from.

This chapter covers some of the history, capabilities, and future of existing and emerging standards for encoding HDR images. We focus here on the bit encodings for each pixel, as opposed to the file wrappers used to store entire images. This is to avoid confusing color space quantization and image compression, which are, to some extent, separable issues. We have plenty to talk about without getting into the details of discrete cosine transforms, wavelets, and entropy encoding. Specifically, we want to answer some basic questions about HDR color encodings and their uses.

What Is a Color Space, Exactly?

In very simple terms, the human visual system has three different types of color-sensing cells in the eye, each with different spectral sensitivities. (Actually, there are four types



GPU Gems

GPU Gems is now available, right here, online. You can purchase a beautifully printed version of this book, and others in the series, at a 30% discount courtesy of InformIT and Addison-Wesley.

Please visit our Recent Documents page to see all the latest whitepapers and conference presentations that can help you with your projects.

Chapter 26. The OpenEXR Image File Format

Florian Kainz
Industrial Light & Magic

Rod Bogart
Industrial Light & Magic

Drew Hess
Industrial Light & Magic

Most images created on a GPU are fleeting and exist for only a fraction of a second. But occasionally you create one worth keeping. When you do, it's best to store it in an image file format that retains the high dynamic range possible in the NVIDIA `half` type and stores additional data channels, as well.

In this chapter, we describe the OpenEXR image file format, give examples of reading and writing GPU image buffers, and discuss issues associated with image display.

26.1 What Is OpenEXR?

OpenEXR is a high-dynamic-range image file format developed by Industrial Light & Magic (ILM) for use in computer imaging applications. The OpenEXR Web site, www.openexr.org, has full details on the image file format itself. This section summarizes some of the key features for storing high-dynamic-range images.

26.1.1 High-Dynamic-Range Images