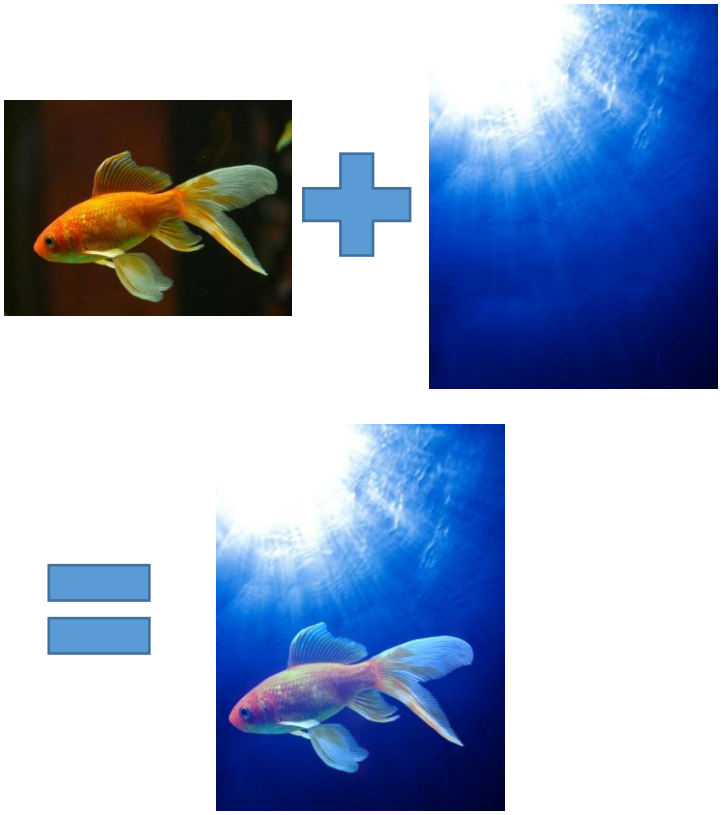


POISSON IMAGE EDITING

An intuitive approach to the second derivative

ABSTRACT

Poisson Image Editing is a method of seamlessly blending one digital image into another. The theory and concepts that are at the core of this process are rooted deeply in calculus, with a particular focus on the second derivative of a function. This lesson will guide students through some of the basics of digital image processing, while providing hands-on experience with industrial software applications that will allow them to write their own image editing programs. The goal of the lesson is for students to develop an intuitive understanding of the significance of the first and second derivatives by being able to visualize their effects on digital images.

Jeremy Moore

Calculus

Contents

LESSON: Poisson Image Blending

Summary

The increasing popularity of digital image viewing and editing software has resulted in a higher demand for methods to manipulate digital images in meaningful and efficient ways. As digital camera technology has advanced, so has image resolution, resulting in higher definition images. In order to keep up with these advances, the field of digital image processing has progressed significantly by combining computer science with some fundamental mathematical principals, many of which are taught at the secondary level, including finding the arithmetic and geometric means, polynomial multiplication and factoring, matrix algebra, probability, exponential and logarithmic equations, calculating distances, and differential equations.

Engineering Connection

Digital images are stored as matrices in which each element in the matrix is a numerical value representing the color or grayscale intensity at that "point" in the image. Each of these numerical values represents a pixel. As a matrix, mathematical operations can be performed on subgroups of elements in order to alter the image in some way.
In this lesson students will explore the concept of Poisson Image Editing, which causes the effect of seamless blending of images. The mathematical concepts that software and electrical engineers must know for this are:

- Matrix Algebra
- Solving Systems of Equations
- First and Second Order Differentiation

Subject Area

Calculus

Time Required

- Lesson: 180 minutes
  - Plan A: 180 minutes
  - Plan B: 135 minutes
  - Plan C: 60 minutes
- Introduction and Module 1: 45 minutes
- Module 2: 45 minutes
- Module 3: 60 minutes
- Module 4: 30 minutes

Grade Level

11 & 12

Standards

Mathematics Florida Standards (MAFS)

MAFS.912.A-REI.3.9 Find the inverse of a matrix if it exists and use it to solve systems of linear equations (using technology for matrices of dimension 3 × 3 or greater).
MAFS.912.N-VM.3.6 Use matrices to represent and manipulate data, e.g., to represent payoffs or incidence relationships in a network.
MAFS.912.N-VM.3.8 Add, subtract, and multiply matrices of appropriate dimensions.
MAFS.912.C.2.1 Understand the concept of derivative geometrically, numerically, and analytically, and interpret the derivative as an instantaneous rate of change or as the slope of the tangent line.
MAFS.912.C.2.2 State, understand, and apply the definition of derivative.
MAFS.912.C.2.8 Find second derivatives and derivatives of higher order.
MAFS.912.C.3.3 Decide where functions are decreasing and increasing. Understand the relationship between the increasing and decreasing behavior of f and the sign of f'.
MAFS.912.C.3.6 Use first and second derivatives to help sketch graphs. Compare the corresponding characteristics of the graphs of f, f', and f".

Materials

Computers with Octave.
Worksheets.

Keywords

concavity, image blending, image processing, matrices, Octave, Poisson, second derivative

Instructional Design Framework

Direct Instruction, Guided Inquiry, Cooperative Learning, Discovery Learning

Learning Objectives

After this lesson, students should be able to:

- Explain how digital images are stored.
- Plot pixel intensity values in one dimension.
- Graph the first and second derivatives as functions.
- Understand how the concavity of the intensity function relates to image sharpness.
- Realize that by matching the concavity of two images, they can be blended.

Prerequisite Knowledge

- Plotting relationships in two dimensions.
- Solving systems of linear equations with two and three variables.
- Familiarity with the concept of a derivative as a rate of change.

Vocabulary

Term	Definition
convolution	A function derived from two given functions that expresses how the shape of one is modified by the other.
GUI	A **graphical user interface** is a human-computer interface (i.e., a way for humans to interact with computers) that uses windows, icons and menus and which can be manipulated by a mouse (and often to a limited extent by a keyboard as well).
image processing	The analysis and manipulation of a digitized image, especially in order to improve its quality.
Integrated Development Environment (IDE)	A set of programming tools for writing applications, all activated from a common user interface and menus. IDEs are necessary standard procedure for program development.
interpolation	A method of constructing new data points within the range of a discrete set of known data points.
pixel	From “picture element”. A minute area of illumination on a display screen, one of many from which an image is composed.

LESSON BACKGROUND & CONCEPTS FOR TEACHERS

Introduction and Motivation

Digital images are immensely popular. Everywhere you go people are snapping away, capturing moments that they can enjoy or share with others over and over again. This popularity has given rise to numerous software programs and applications that allow users to modify their images in many ways. We are able to resize, rotate, change contrast, enhance colors, and add different effects to images with a few mouse clicks. The methodology that is occurring "behind the scenes", i.e. in the programming, may seem to be unattainable by many. By introducing students to the basics of image processing in high school, they can experience firsthand that many of the techniques that are used in these programs are based on the fundamental concepts that they are learning about in their math classes. In fact, within a single class period they will be able to write basic programs that accomplish some of the same outcomes as their favorite image editing programs.

What is a Digital Image?

A digital image is interpreted as a rectangular grid in which each element is a pixel. The pixels are identified by their location in the grid, similar to the way coordinates on a graph are identified. For an image, the "origin" is the top left pixel. For our purposes, we will call this (1, 1). The convention for identifying pixel location comes from the standard notation for elements in a matrix, which is (row, column). Thus, the origin pixel is in the first row, first column, and pixel (3, 2) is in the third row, second column. Take the 3 by 3 matrix below. If we view this as a function, say f, we would identify $f(1,1) = 255, f(2,3) = 26$, and $f(3,1) = 0$.

$$\begin{bmatrix} 255 & 200 & 135 \\ 52 & 101 & 26 \\ 0 & 58 & 187 \end{bmatrix}$$

Now that we have a way to identify each pixel, we can investigate what the value stored at that location represents. Each pixel stores a single numerical value that represents the intensity of the image at that location. This applies to color images as well as black-and-white (grayscale). In a grayscale image, the intensity range goes from black (lowest) to white (highest). With color images, it is a little more involved. There are many different methods for representing color in an image. The most common method, which is the one that we will adopt, is red-green-blue (RGB). The idea is that by combing different intensities of only these three colors, any color in the visible spectrum can be produced. We notice in figure 1 that the combination of all three colors results in white, while the absence of the colors is black.

Figure 1

RGB images are digitally stored in matrices with three dimensions, so the size would be width by height by 3. The third dimension represents the color *channel,* which is red, green, or blue.

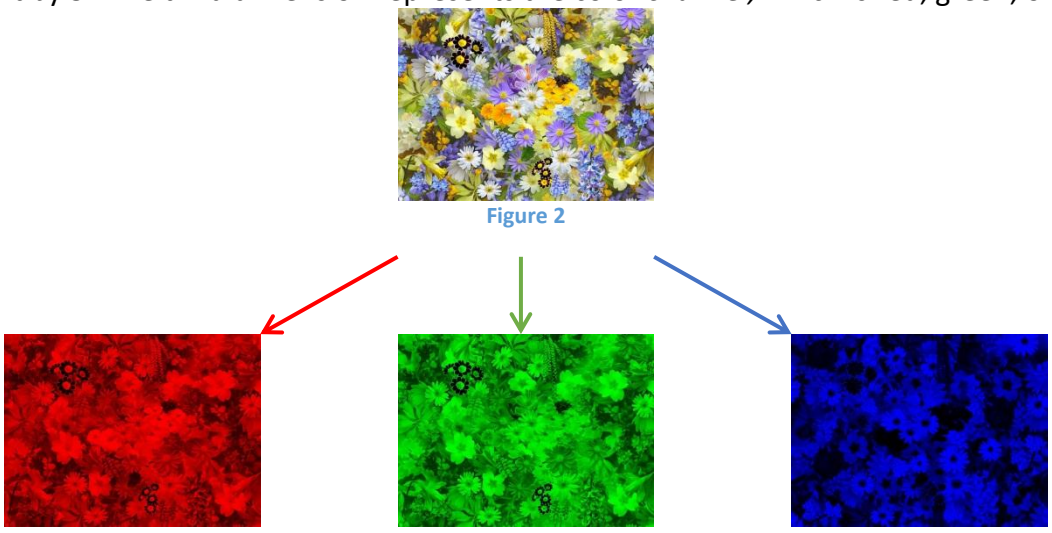


Figure 2

In figure 2 we have separated the three channels to visualize the intensity values that combine to make the original image. Digitally, each of those is a matrix. So the any pixel at a location (row, column) of the original image is actually the combination of the pixels at that location in the three channels below. The red intensity is represented by pixel (row, column, 1), green is (row, column, 2),and blue is (row, column, 3).
There are many ways to represent the scale of intensities for each pixel, the most common being 0 to 255. The reason for this has to do with the fact that all data is represented in binary form at the computational level. A single byte is comprised of eight bits, each of which is either a 0 or 1, so there are 2^8 different bit combinations for any byte, and in the context of images, each represents a different intensity.

In some cases it is advantageous to scale the intensities from 0 to 1. In this case, 0 would mean that none of the intensity is present, while 1 indicates that it has the full intensity. In grayscale images, 0 is black and 1 is white. For color images, if all three channels are 0, none of the color intensities are present, and if all three channels are 1, the full intensity of each color is present.

With the numerical values of each pixel stored as a matrix, the image can be manipulated using mathematics. This concept is at the heart of image processing, and the results of what can be accomplished are limitless. In this lesson we will focus on the way in which pixel values change from one to the next, which represents an approximation of the derivative.

Using Calculus with Image Processing

By representing a digital image as a matrix, it is possible to view the pixel values as a continuum. From this perspective we can determine the rate of change from one pixel to the next, which is an approximation of the first derivative. By determining the rate of change of those values, we can approximate the second derivative and concavity.

We begin with a matrix representation of figure 3 in which the pixel values have been scaled to be between zero and one.

Figure 3

$$\begin{bmatrix} 0.25948 & 0.22027 & \cdots & 0.68985 & 0.73299 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0.22672 & 0.23064 & \cdots & 0.15243 & 0.16027 \end{bmatrix}$$

By isolating a single row of the matrix, the sequential values represent a single variable function. In figure 4, row 50 has been chosen.

Figure 4

$$[0.130306 \;\; 0.130306 \ldots 0.218707 \;\; 0.226550]$$

The values from row 50 can be plotted in a Cartesian graph as we would for any function. Figure 5 shows the pixel samples and graph of the pixel values in this row. Notice how higher values in the graph correspond with lighter tones in the image.

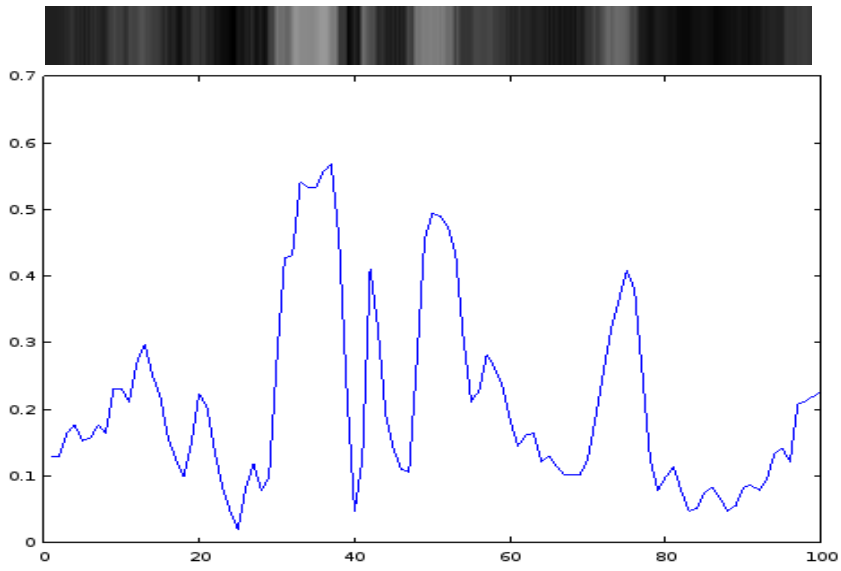


Figure 5

By viewing the pixels as a function, it is possible to quantify the change in pixel intensity in the context of a derivative. Using the continuous definition of the derivative, we can set Δx to equal one pixel to arrive at our derivative approximation for the discrete case.

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

$$\approx f(x + 1) - f(x).$$

This derivative approximation can be applied to the values in row 50 to quantify the rate of from pixel to pixel. This can be seen in figure 6. Notice that positive values correspond to lightening trends, while negative values show a darkening. Likewise, the greater the magnitude, the more drastic the intensity change.

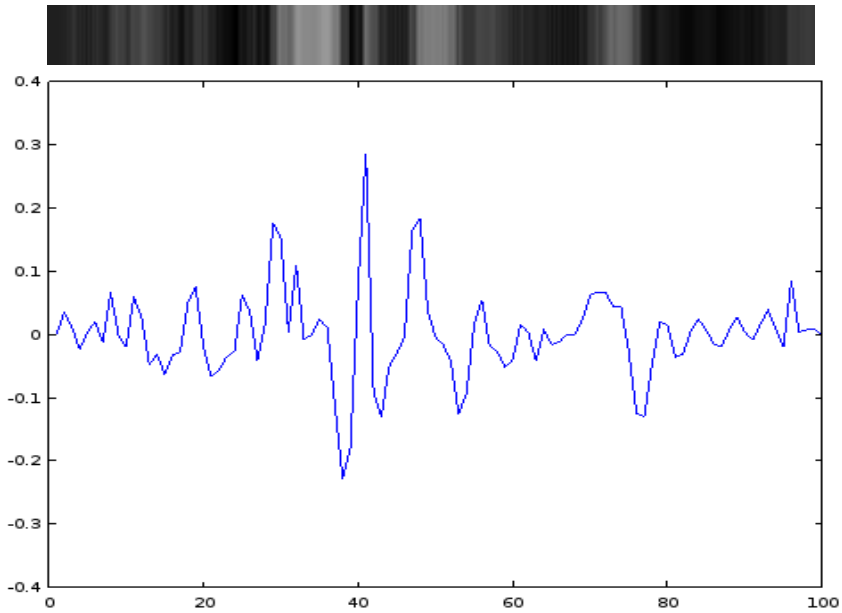


Figure 6

In a similar manner, the second derivative can be approximated by finding the difference between the rate of change leading up to and away from any pixel, resulting in

$$f''(x) \approx [f(x + 1) - f(x)\,] - [f(x) - f(x - 1)\,]$$

$$= f(x + 1) + f(x - 1) - 2f(x).$$

By taking a closer look at a section of row 50 from pixel 30 to 60, the dynamics of the graphs become clearer. In figure 7 we can see the relationships that exist between the function and its derivatives, including critical points, relative extrema, points of inflection, and concavity.

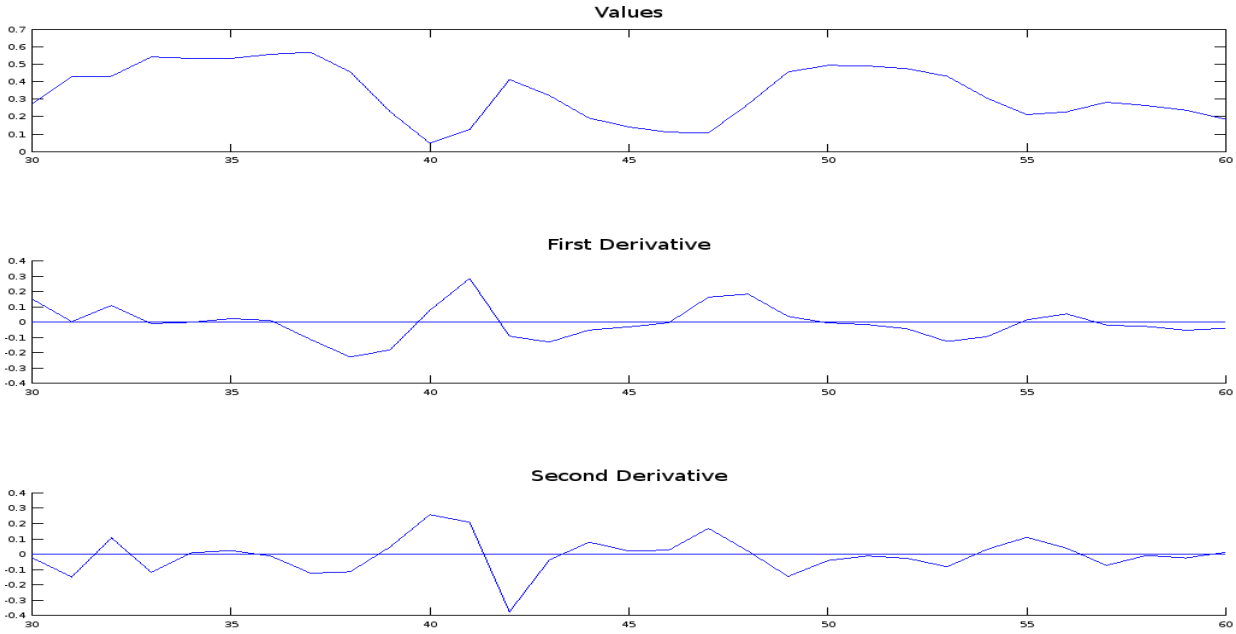


Figure 7

The Laplacian Operator

This process of calculating first and second derivative approximations can be repeated for every row in the matrix, as well as every column, which tells us how the intensity values are changing horizontally and vertically at each pixel. By knowing this, it is possible to identify sharp changes in the image, which usually represents the outlines of individual objects.

By taking the perspective of the image as a function f, and the rows and columns as x and y axes, a pixel neighborhood around $f(x,y)$ would look like figure 8.

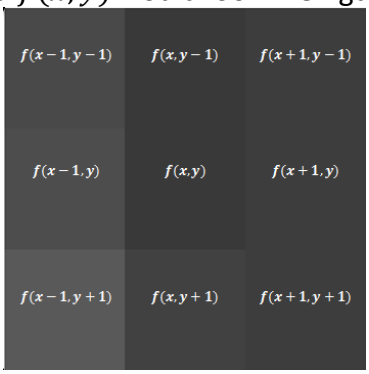


Figure 8

The focus of this lesson deals with the Laplacian operator, ∇^2, which can be found at a given point by adding the second derivative approximations along the row and column. We use partial derivative notation since there are two input values, but it simplifies very nicely using the discrete definition from above.

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$= f(x+1,y) + f(x-1,y) - 2f(x,y) + f(x,y+1) + f(x,y-1) - 2f(x,y)$$

$$= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y).$$

We can use figure 9 as an example of how to calculate the Laplacian for a single pixel.

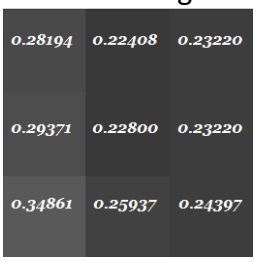


Figure 9

$$\begin{aligned}\nabla^2 f &= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y) \\ &= 0.23220 + 0.29371 + 0.25937 + 0.22408 - 4(0.22800) \\ &= 0.097360\end{aligned}$$

Since the Laplacian is a second derivative operator, its value represents concavity. Portions of the image for which the Laplacian is negative, the intensity values would be concave down, and thus have a "peak" of brightness somewhere in that region. The opposite occurs if it is positive. One useful outcome of this observation is that if the Laplacian is subtracted from the original function, it "enhances" changes in the function, which can be observed in figure 10.

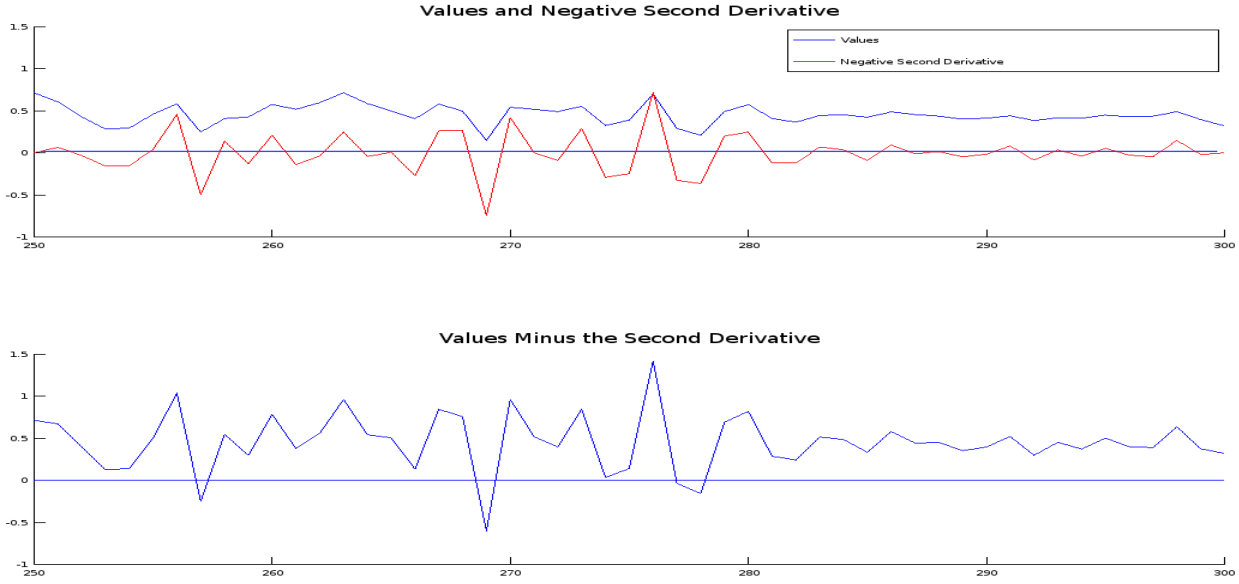


Figure 10

In the context of an image, subtracting the Laplacian causes a sharpening effect. Figure 11 is slightly blurred. By calculating the Laplacian at each pixel and subtracting it, the result is figure 12, in which some of the image's definition has been restored. The script used to create this effect is available here.

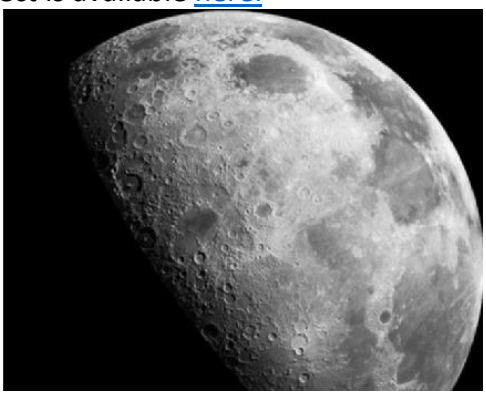


Figure 11

Figure 12

Image Blending

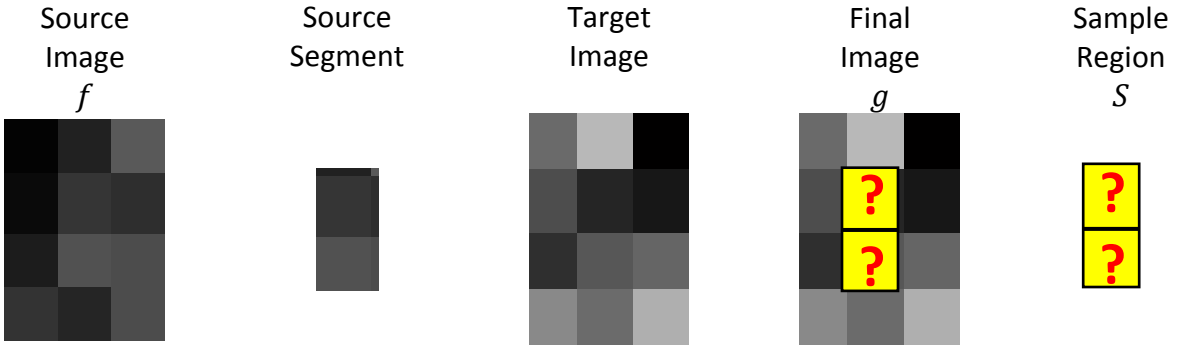
To seamlessly blend one image into another, we can use the process known as Poisson Image Editing. The method is named after the French mathematician and physicist Siméon Denis Poisson, since it is based on solving Poisson equations. The process involves a *source image* to be blended into a *target image*, which combine to create the *final image*, which can be seen in figure 13.

Source Image **Target Image** **Final Image**

Figure 13

The idea is that the values of the Laplacian for the pixels in the source should be as close as possible to the Laplacian for the pixels in the same portion of the final image. Thinking in terms of concavity, if the final image has the same boundary as the target image, and the same concavity as the source image, so the shape of the source image should be preserved within the context of the target.

Figure 14 illustrates the process of Poisson Image Editing in a simple case in which a two-pixel segment from a source image is to be blended into a twelve-pixel target image. This example does not necessarily demonstrate the visual effect, since the image is so small, but is sufficient in showing the methodology.



$$f = \begin{bmatrix} 0.012 & 0.133 & 0.351 \\ 0.041 & 0.211 & 0.180 \\ 0.116 & 0.325 & 0.304 \\ 0.200 & 0.159 & 0.302 \end{bmatrix} \qquad g = \begin{bmatrix} 0.417 & 0.720 & 0.000 \\ 0.302 & ? & 0.092 \\ 0.186 & ? & 0.396 \\ 0.538 & 0.419 & 0.685 \end{bmatrix}$$

Figure 14

To find the missing values in S, we will need to solve a system of Poisson equations. This means that the Laplacian from the source segment must equal the Laplacian in the final image for the values in S, or

$$\nabla^2 f(2,2) = \nabla^2 g(2,2) \tag{1}$$

and

$$\nabla^2 f(3,2) = \nabla^2 g(3,2). \tag{2}$$

These calculations are easily accomplished since we have already established a discrete form for the Laplacian,

$$\nabla^2 f = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y).$$

Equation 1 above becomes

$$0.180 + 0.041 + 0.325 + 0.133 - 4(0.221) = 0.092 + 0.302 + g(3,2) + 0.720 - 4\big(g(2,2)\big)$$

$$-0.165 = 1.114 + g(3,2) - 4\big(g(2,2)\big)$$

$$4\big(g(2,2)\big) - g(3,2) = 1.279. \tag{3}$$

Equation 2 becomes

$$0.304 + 0.116 + 0.159 + 0.211 - 4(0.325) = 0.396 + 0.186 + 0.419 + g(2,2) - 4\big(g(3,2)\big)$$

$$-0.510 = 1.001 + g(2,2) - 4\big(g(3,2)\big)$$

$$-g(2,2) + 4\big(g(3,2)\big) = 1.511. \tag{4}$$

Equations 3 and 4 make a linear system that can be solved easily. We find that

$$g(2,2) = 0.4418$$
$$g(3,2) = 0.4882.$$

Figure 15 shows the result of blending, which has preserved the concavity of the source image within the boundaries of the target.

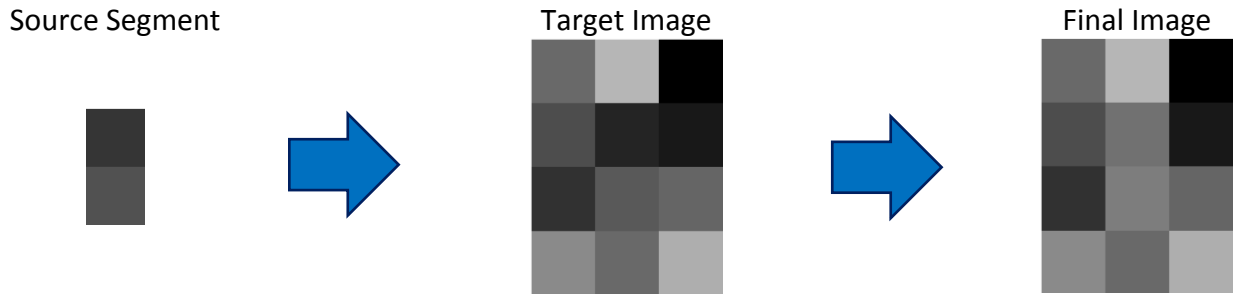


Figure 15

In this trivial case, we do not have enough pixels to visualize what the blending has accomplished to the full extent. With Poisson Image Editing, the number of pixels in the source segment is equal to the number of equations that need to be solved. To demonstrate this on a large scale would be overly cumbersome, though programs like MATLAB and Octave can perform them relatively quickly. At this point, students can use the provided Image Blending Demo themselves to blend two of their own images.

The Concepts

1. Digital images are stored as matrices of pixel intensities.
2. Each pixel value is stored at a specific "coordinate".
3. By viewing these values sequentially, we can calculate the rate of change from pixel to pixel, giving an approximation of the derivative in one direction.
4. The rates of change of the derivative approximations provide approximate values of the second derivative, which determines concavity.
5. The first and second derivative approximations express how the pixel intensities transition from one to the next, meaning that the image darkens or brightens.
6. The second derivative in two directions is called the Laplacian.
7. The Laplacian provides insight into the sharpness of the image.
8. Poisson Image Editing causes a seamless blending of images by setting the Laplacian of a source image equal to the Laplacian of a final image for which the boundary values are determined by a target image.
9. Programs can be written using mathematical software tools such as MATLAB or Octave that perform operations across the rows and columns of an image.

LESSON PROCEDURES

- **Before the lesson**
  - Ensure that Octave has been installed on the computers that students will be using.
  - Gather materials, make copies of handouts/worksheets
  - Introduction & Motivation
    - Give Background (See below)
    - PPT1
- **During the lesson**
  - Module 1 – Getting to Know Image Processing with Octave
    - Students will learn the structure of the Octave IDE, including basic commands, relevant syntax, and image processing tools.
    - PPT2
    - Worksheet1
    - Activity1 – How to Get the Negative of an Image
  - Module 2 – The Calculus of Images
    - Students will isolate a single row of pixels from their image.
    - They will plot the pixels values sequentially by their intensity values.
    - Then, they will calculate first and second derivatives, and plot them as well.
    - Students will be able to explain what each of the graphs represent in the context of the original picture.
    - Using the first and second derivatives, students will be able to determine where there are sharp changes in intensity within a section of the image.
    - PPT3
    - Activity2
    - Image Analysis Report
    - Image Analysis Report (sample)
  - Module 3 – Change in Two Dimensions
    - Students will extend their knowledge from Module 2 to analyze change along rows and columns. These values will determine the outlines of individual objects within an image.
    - Students will be introduced to the Laplacian Operator, which will determine the second order partial derivative approximations in two dimensions.

        - Students will be able to use these values to sharpen the definition in unclear images.
        - PPT4
    - Module 4 – Image Blending
        - Using the second derivative along the rows and columns of an image, students will observe the mathematical concept behind Poisson Image Editing. This is a method in which an image is superimposed onto another image, and they are "seamlessly" blended into a single image.
        - Students will understand that by matching the concavity of the two images, they will smoothly transition into one another.
        - Students will see that the image can be created by solving a linear system of equations.
        - PPT5
- **After the lesson**
    - Students will complete a summative assessment, which includes using Octave.
    - Student will provide feedback in an exit survey.
    - Teachers will analyze student feedback and modify the lesson as needed for future implementation.

Introduction & Motivation

(To the students) How many of you have ever taken a picture using a digital camera, smartphone, or tablet? (Everyone raises their hands)
How many of you have ever used a computer program or app to change something about the image? (Again, we should see universal response)
What are a few ways that you could change an image? (Responses should include resize, enhance color, crop, etc…)
What are some programs that you have used for this? (Answers should include Photoshop, MS Paint, Instagram, etc…)
Does anyone have any idea of how these programs work? (Minimal response expected)
How many of you think that it is too complicated to understand? (Mixed responses, hopefully)
Today you are going to see that it is not at all beyond your capabilities to understand, or even write some of the programs that you have used many times on your own digital images. (This should hopefully evoke a skeptical interest in what's to follow)

To begin, we have to understand how images are stored digitally. Who can tell me the fundamental components that every digital image is composed of? (At least one person, if not many, should know that it is a pixel.)

The way that these pixels are stored is in a grid, which we refer to as a matrix or array. Each pixel is a number that represents the brightness, or intensity, of the picture at that exact point in the picture. Each pixel has a coordinate that represents its row and column, starting from the top left with pixel (1,1).
(Show digital image to matrix demonstration)

Once the image has been converted to a matrix, we have the power to change its appearance in any way that we want by changing the values of the pixels. In order to do this, we will need to learn a little bit about some mathematical software applications, and with a few simple commands we can make our own image editing programs. We are going to use an Integrated Development Environment (give brief explanation of IDE) called Octave. This is a clone of a program called MATLAB, which is an extremely powerful industry tool used by engineers around the world.

Accompanying Material: PPT1 and Worksheet1

Lesson Scaling

- **Implementation and Student Level Scaling**
  - **Plan A:** Full implementation of all four modules of this lesson plan including hands-on activities working with Octave and teaching of the lesson content. This may take 4 – 5 days and include significant lab time.
  - **Plan B:** Full implementation of Modules 1 and 2, and a demonstration of Modules 3 and 4.
  - **Plan C:** Demonstration only. An abbreviated explanation of Module 1, and a more elaborate explanation of Modules 3 and 4.

ACTIVITY 1: How to Get the Negative of an Image

Activity 1 Goals / Background Information

The negative of an image is one in which lighter colors appear darker, and darker colors appear lighter. This is accomplished by subtracting each pixel's intensity from the maximum possible intensity. The final result will be for students to produce the negative of one of their own images.

Activity 1 Procedures

The procedures are detailed in steps on the handout.

Activity 1 Answers

Students will be using their own images, so answers will vary. The goal is for them to understand how images are represented and processed in Octave.

Activity 1 Extension

Students who pick up on this quickly may proceed onto the Activity 1 Extension, in which they will alter the image contrast by using exponents.

ACTIVITY 2: Calculating Change in an Image

Activity 2 Goals / Background Information

The pixel intensities in an image can be mapped as a function. In this activity, students will isolate a single row in an image, and plot the pixel intensities as a function, and plot its first and second derivatives by calculating the rate of change in pixel intensity. They should be able to identify critical points and points of inflection, and explain their meaning in terms of the context of an image.

Activity 2 Procedures

The procedures are detailed in steps on the handout.

Activity 2 Answers

At the completion of the activity students will complete an Image Analysis Report. Students will be using their own images, so answers will vary. A sample report is provided for the teacher and student get an idea of what is expected. A sample script is provided for the teacher.

ATTACHMENTS

- PowerPoints for lecture
  - PPT1
  - PPT2
  - PPT3
  - PPT4
  - PPT5
- Handouts
  - Worksheet1 (Word/PDF)
  - Activity1 (Word/PDF)
  - Activity1 Extension (Word/PDF)
  - Activity2 (Word/PDF)
  - Image Analysis Report (Word/PDF)
  - Sample Image Analysis Report (Word/PDF)
- Octave code
  - Activity2Script
  - PowerPoint4Slide14Script
- Video
  - Plot3D_1
- Images
  - chimp
  - moon

REFERENCES

1. http://pixabay.com/en/veiltail-fish-goldfish-swim-11451/
   Public Domain CC0 1.0
2. http://pixabay.com/en/below-beneath-blue-deep-dive-19038/
   Public Domain CC0 1.0
3. http://pixabay.com/en/intersection-mix-colors-rgb-red-154782/
   Public Domain CC0 1.0
4. http://pixabay.com/en/spring-flowers-flowers-collage-110671/
   Public Domain CC0 1.0
5. Original image taken by Jeremy Moore
6. http://pixabay.com/en/animal-ape-chimp-chimpanzee-cute-2602/
   Public Domain CC0 1.0
7. http://pixabay.com/en/moon-half-moon-light-silhouette-322222/
   Public Domain CC0 1.0

CREDITS

Authors and Contributors:

Jeremy Moore, Teacher – Brevard Public Schools, Brevard County, FL

Date created / updated:

Created July 3, 2014. Revised July 15, 2014.

Supporting Programs:

AEGIS RET Program, College of Engineering and Computer Science, University of Central Florida, and College of Engineering, Florida Institute of Technology

Acknowledgements:

This curriculum was developed under National Science Foundation RET grant #1200566 and #1200552. However, these contents do not necessarily represent the policies of the National Science Foundation, and you should not assume endorsement by the federal government

Contact information:

Jeremy Moore (moore.jeremy@brevardschools.org)

Copyright: