Math 2590 Podcast Script
Tuesday, November 30, 2010
Re-Submitted to: Mike Zabrocki
Submitted by: Mary Michael, Sharla Niroopan & Jocelyn Santos

1. Hello, and welcome to our podcast for MATH 2590.

2. Before we start, we need you to smile for the camera: 3, 2, 1 *shutter sound effect or group of people saying "cheese"*

3. Beautiful! You may notice that with newer digital cameras, that prior to snapping a picture it automatically focuses on the faces of your friends and family, placing a rectangle around each face. Have you ever wondered how the camera knows where the faces are? How does it determine this?

4. Have you uploaded an album onto Facebook recently? You'll notice that when you tag one picture, Facebook uses software to find faces and automatically tags them in all pictures of that album. It's an amazing time saver. Gone are the days of tagging each individual picture. How does Facebook do this?

5. Are you a gamer? Have you had a chance to play with Kinect? Kinect is the latest gaming system that does not require a remote. Kinect has a system of sensors that can detect and respond to the location and movements of your face and body.

6. So what do digital cameras, Facebook's photo application and Kinect have in common? These three things are linked because they are all able to find a human face. But, how, you might ask?

7. Well, this is where the mathematics of facial detection software comes into play.

8. Let us walk you through this sophisticated mathematical system that we encounter all the time.

9. Face detection is a computer technology that determines the locations and sizes of human faces. It detects facial features and ignores anything else in the background, placing a rectangle around each face. In order for this to be successful, 2 conditions must be met:
   a. One: There must be enough light.
   b. Two: The face must be in full view, right side up, and directly facing the camera lens.

10. If these two criteria are met, the software is able to translate the captured information into a black and white image.

11. At this point the program is only interested in large black areas of a certain size and shape, and will disregard or ignore any black areas that are too small to be potential facial features. Now the facial detection software works to verify the presence of individual facial features, namely the eyes, nostrils, mouth and eyebrows.

12. The software relies upon a standardized algorithm that defines the typical proportions of most faces. An algorithm is an effective method for solving a problem, carried out through a sequence of steps. You can think of it as a list of instructions for completing a task, like a recipe.

13. The algorithm used for this system defines the expected proportions between the features of a typical face. The software then measures the distance between two potential facial features at a time, and uses the algorithm to confirm if this distance measured matches the expected proportions.

14. For example, if the algorithm states that the proportion between two eyes must be X, then the measured distance between the two potential facial features must be X or a multiple of X to be classified as two eyes.

15. The software continues this process until it has identified the eyes, nostrils, mouth and eyebrows. Provided all measurements closely match the standard facial proportions, the software can confirm the presence of a face.

16. In other words, in a fraction of a second your digital camera's facial detection software runs this process and finds your family and friends' faces before you snap your picture. *shutter sound effect*

17. So now that we understand how facial detection software works, we see how it can be useful for auto-focus when taking photographs, to save time when tagging pictures on Facebook, and to enjoy remote-free gaming systems.

18. Now, let's think about other applications for this software. Imagine a television that conserves energy using facial detection software. Since many people multitask while their t.v. is on, developers are introducing this software to reduce brightness when you look away, but return to standard view when your face

returns to the screen. This is a great way to apply mathematics to conserve energy in our quest for sustainability.

19. It’s incredible that the mathematics of facial detection software is such a common part of our lives, and can have profound effects on the world we live in.

EXERCISES

Facial detection software measures the distances between potential facial features and compares these measurements to the standard proportions defined by the algorithm.

In Part 1 and Part 2 of these exercises, you will work with the algorithm and practice the calculations performed by facial detection software. In order to prepare you for Part 2, Part 1 allows you to familiarize yourself with the algorithm's process of comparing measurements to standard proportions.

You must be able to compute values using a standard algorithm $D=\sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$ and a set of given variable values.

Part 1

1. Determine the value of D, using the standard algorithm $D=\sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$

    a. $X_1=2$, $X_2=5$, $Y_1=6$, $Y_2=5$

    b. $X_1=2$, $X_2=4$, $Y_1=5$, $Y_2=4$

2. Determine the distance from the eyes to each facial feature, based on the given proportional values indicated.

    a. $D=5$, $D_{nose}=0.45D$, $D_{eyebrows}=0.25D$, $D_{mouth}=1.2D$

    b. $D=3.9$, $D_{nose}=0.5D$, $D_{eyebrows}=0.33D$, $D_{mouth}=0.9D$

3. Determine the value of each proportional value if the measured distance from the eyes to the facial feature is known.

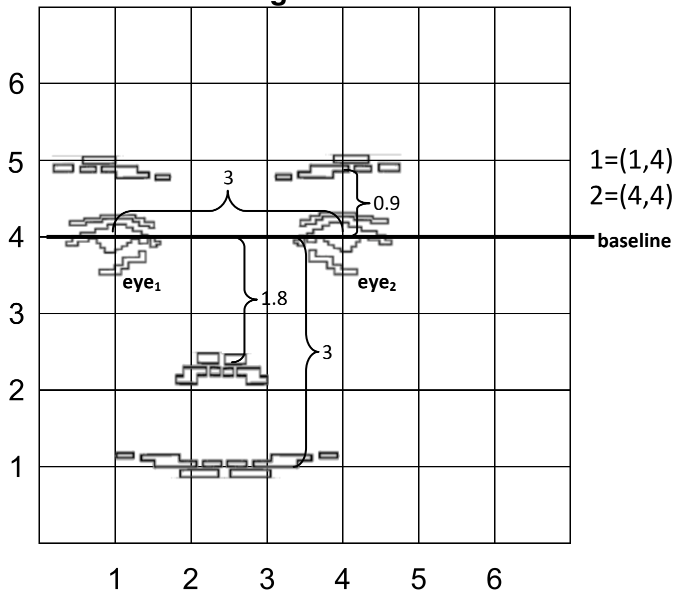
    a. If the distance from the eyes to the nose is 1.68, determine the value of $D_{nose,}$ if $D=4$.

    b. If the distance from the eyes to the eyebrow is 0.75, determine the value of $D_{eyebrow,}$ if $D=3.5$.

Part 2

The standard algorithm used in face detection software must first use a particular base line marked by two particular coordinates to distinguish where the rest of the face's features are located. These particular coordinates are located at the centre of the two eyes, namely (x_1, y_1) at the center of eye 1 and (x_2, y_2) at the center of eye 2. A straight line is then used to connect the two points. This is the baseline. The distance between these two central eye points is calculated by plugging in the coordinates of these two eyes into the standard algorithm, $D=\sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$, where D is used to calculate the distances of D_{nose}, $D_{eyebrows}$, D_{mouth} proportionally from the base line. That is, $D_{nose}=0.6D$, $D_{eyebrows}=0.3D$, $D_{mouth}=1.0D$. Refer to Figure 1 to understand how the baseline is used to locate the nose, eyebrows and mouth.

Figure 1



Sample Calculations:

Distance between the two centres of the eyes:

$D=\sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$

$=\sqrt{(1-4)^2 + (4-4)^2}$

$=\sqrt{9+0}$

$= 3$ ← distance between 2 centres of the eyes

$D_{nose}=0.6D$

$=(0.6)(3)$

$=1.8$

$D_{eyebrows}=0.3D$

$=(0.3)(3)$

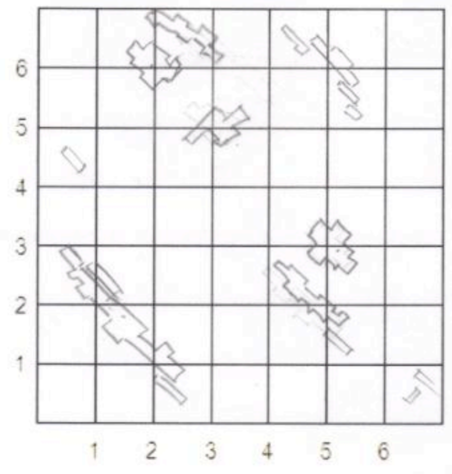
$=0.9$

$D_{mouth}=1.0D$

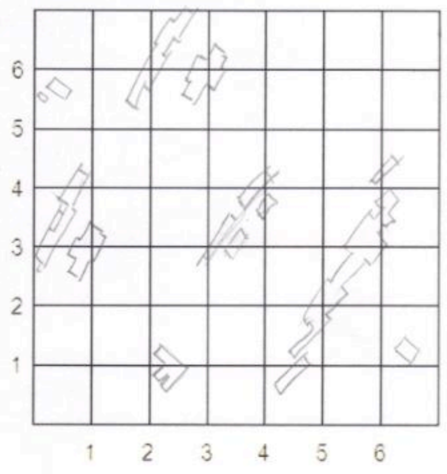
$=(1.0)(3)$

The images below represent binary images used to measure the distances between the eyes, nose, eyebrows and mouth proportionally, where $D_{nose}=0.6D$, $D_{eyebrows}=0.3D$, and $D_{mouth}=1.0D$. Determine which blocks represent each facial candidate, when D=4. To solve this, you must first locate the eyes and establish the baseline. Following this, you must use the baseline, the given value of D, and $D_{nose}=0.6D$, $D_{eyebrows}=0.3D$, and $D_{mouth}=1.0D$ to identify which blocks are the rest of the facial features. Show your work and colour code the blocks (e.x. blue block= nose).

a)



b)



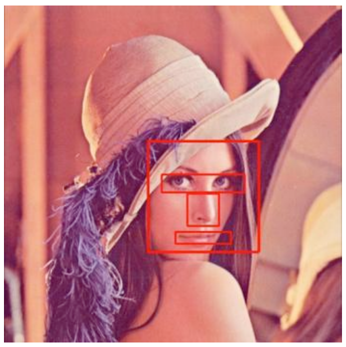
D=4

Part 3
One of the versions of face detection software works by detecting the lips first because they are centered horizontally on the face and are isolated (assuming no facial hair). It then uses the width of the lips to determine the expected locations of the other facial features.

Below are some steps that can be used to obtain a similar result as the software when locating the possible positions of facial features:

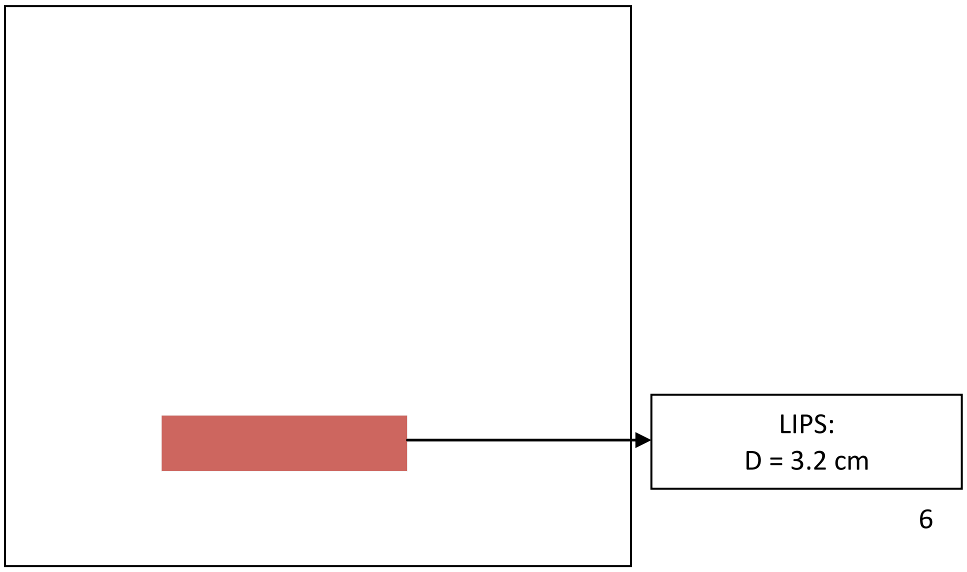
- Firstly it measures the width of the lips (in this example we will label this width as D)
- The width of the lips is equalled to the vertical distance between the bottom of the lips and the bottom of the eyes. Use a ruler to draw a horizontal line that represents where the bottom of the eyes would be located (this is called the base line).
- Next determine what the value of 0.6D is. This is the distance from the center of the base line to the bottom of the nose.
- Draw a horizontal line that measures 0.8D where the bottom of the nose should be located (this represents the width of the nose).
- Draw a horizontal line that is 0.4D above the base line. This is the location of the eyebrows.

Below is an example of what the result of the facial detection software would look like



Assume that the highlighted rectangle below is the detection of lips on a face. Use the above theory to draw the blocks that would represent where the following features would approximately be found on a human face:

a. the nose
b. the eyes
c. the eyebrows



Works Consulted

Jeng, Shi-Hong; Liao, Hong-Yuan Mark, *et al*. “Facial Feature Detection Using Geometrical Face Model: An Efficient Approach” in *Pattern Recognition*. (1996).

Hjelmas, Erik and Low, Boon Kee. “Face Detection: A Survey” in *Computer Vision and Image Understanding*, Vol 83, Issue 3. (September 2001), pp. 236-274.

Rowley, H.A.; Baluja, S.; and Kanade, T. “Neutral network-based face detection” in *Pattern Analysis and Machine Intelligence*, Vol 20, Issue 1. (January, 1998), pp. 23-38.

Yang, Guangzheng & Huang, Thomas S. “Human Face Detection in a Complex Background” in *Pattern Recognition*, Vol. 21, Issue 1. (January, 1994), pp 53-63.