Homework #1

Displays, Image Processing,

Affine Transformations, Hierarchical modeling

Prepared by: David Jones, Andrew Lewis, and Brian Curless

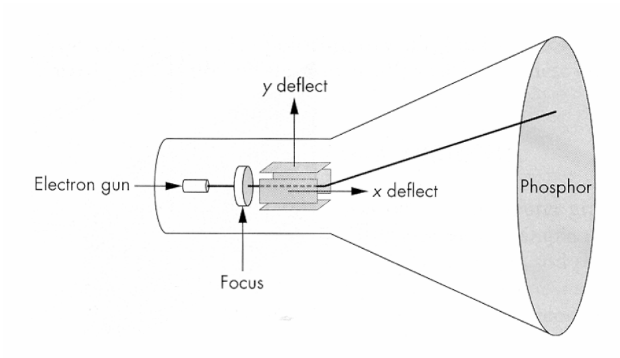
Assigned: Saturday, October 8th
Due: Friday, October 21st

Directions: Please provide short written answers to the questions in the space provided. If you require extra space, you may staple additional pages to the back of your assignment. Feel free to discuss the problems with classmates, but please ***answer the questions on your own*****.**

Name:____

Problem 1: Raster Displays (25 points)

Consider a monochrome (one electron beam, no color) 15" CRT display running at 1024x768, 72 Hz. The width of such a display is 300mm and the height is 225mm. Below is an illustration of a CRT display with a round screen, but you should assume that the screen is actually rectangular. After the electron beam hits the phosphors, they emit light for approximately 50 μs. Typically, the light peaks in brightness and then decays exponentially, but let's assume for simplicity that the light is on for 50 μs and then turn off. For this problem, you can also assume that the retrace or blanking interval – the period when the beam moves to the beginning of a scanline and to the top of the screen – takes essentially zero time. For each sub-problem, **show your work**.



(a) (2 points) How fast is the beam moving across the screen, in meters/second and in kilometers/hour?

(b) (2 points) What fraction of the screen is lit at any moment in time?

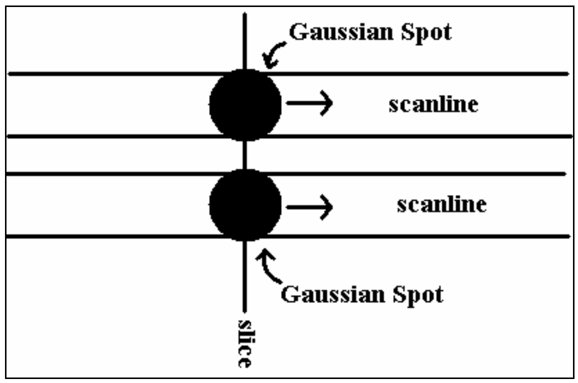
(c) (3 points) How wide is each pixel on the screen, in mm, and how long does it take the electron beam to draw it?

Problem 1: Raster Displays (cont'd)

(d) (8 points) The electron beam of our display has a Gaussian cross-section:

$$G(x,y) = e^{\frac{-\left(x^2+y^2\right)}{2\sigma^2}}$$

The figure below shows the Gaussian spot sweeping along two consecutive scanlines. Two spots are drawn together as the scanlines are swept in parallel, though the sweeps happen one scanline at a time. Our visual system merges the result, effectively adding together all the light over time, so that it seems like all scanlines are drawn simultaneously and additively.



Suppose the σ of the beam is 0.11mm. Now consider the case where one scanline is displaying a constant value of 1.0, and the next scanline is displaying a constant value of 0.5. You may neglect all other scanlines (assume the rest of the screen is black), and you may assume that the display is gamma-corrected (no non-linearities). Take a vertical slice through the beam profile as though two spots, one above the other, were emitting light at the same time as shown in the figure. Write out the equation of the function you would get along that slice and sketch it. You can use a software, e.g., Excel or Matlab, to draw the function.

Problem 1: Raster Displays (cont'd)

(e) (5 points) Suppose that the display is a multi-sync monitor, and that you now drive it at 1600x1200. As it turns out, manufacturers do not allow one to change the width of the electron beam cross-section, so it has the same sigma as before. What will the cross-section in (d) now look like? Are the scanlines going to be well resolved? Sketch the function to justify your answer.

(f) (5 points) Suppose you now drive this multi-sync monitor at 640x480. What will the cross-section now look like? How will the scanlines look on the screen? Sketch the function to justify your answer.

Problem 2: Image Processing (15 points)

Suppose we have two filters:

0	0	0
-1	0	1
0	0	0

Filter *A*

0	1	0
0	2	0
0	1	0

Filter *B*

(a) (3 points) In class, we described a simple and intuitive version of an *x*-gradient filter: [-1 1]. When applied, this filter computes the *finite difference* gradient in the *x*-direction, essentially solving for $\partial f / \partial x \approx \Delta f / \Delta x$, where $\Delta x = 1$ and pixels are one unit distance from their neighbors. Filter *A*, by contrast, is used to compute what is known as the *central difference* *x*-gradient. Although it cannot be normalized in the usual way, since its values sum to zero, it is usually multiplied by a coefficient of ½. Why?

(b) (3 points) Normalize *B*. What will effect will this normalized filter have when applied to an image?

Problem 2: Image Processing (cont'd)

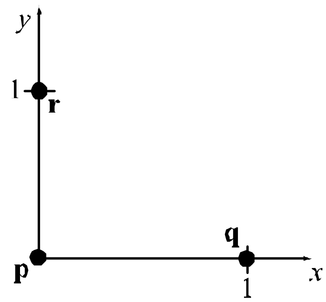
(c) (4 points) Compute $A*B$, using A and B from the ***original*** problem statement, i.e., ***without*** using the scale factors described in (a) and (b). You can treat B as the filter kernel and assume that A is zero outside of its support. You do *not* need to show your work. [Aside: convolution is commutative ($A*B=B*A$), so you would get the same answer by using A as the filter kernel. But, you would have to remember to "flip" the kernel to get $\widetilde{A}[i, j] = A[-i,-j]$. We've asked you instead to use B as the filter kernel, but since B is symmetric, i.e., $\widetilde{B}[i, j] = B[-i,-j] = B[i, j]$, you don't need to worry about flipping.]

(d) (2 points) Compute $A*B$, now using A and B after scaling them according to (a) and (b).

(e) (3 points) If we apply the result of (c) or (d) to an image f, we are computing $(A*B)*f$. Convolution is associative, so we would get the same results as computing $A*(B*f)$. In other words, we're filtering the image with B, and then filtering the result with A. Why would it be desirable to apply B before computing the gradient (as opposed to not applying B at all)? Why might applying B be better than applying a filter B' that is filled with a full 3x3 set of coefficients, rather than just a single column of coefficients?

Problem 3: 2D Affine Transformations (25 points)

Consider the following points on the 2D plane, and an affine matrix M that will transform them:



$$M = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix}$$

(a) (6 points) Suppose we M to leave **p** and **q** fixed; i.e., $M\mathbf{p} = \mathbf{p}$ and $M\mathbf{q} = \mathbf{q}$. What constraints does this put on the elements of M? Justify your answer.

(b) (5 points) Provide two (non-identity) affine transformation matrices that satisfy the constraint in (a), and name them; i.e., choose two different transformations from the possibilities we discussed in class. Will the products of these transformations also satisfy the constraint? Show your work.

Problem 3: 2D Affine Transformations (cont'd)

(c) (3 points) If an affine transformation fixes **p** and **q,** does it also fix any point on the *x*-axis? Justify your answer.

(d) (3 points) If we only wanted to fix **p**, i.e., $M\mathbf{p} = \mathbf{p}$, what constraints would be placed on the elements of M? What general class of transformations satisfies these constraints?

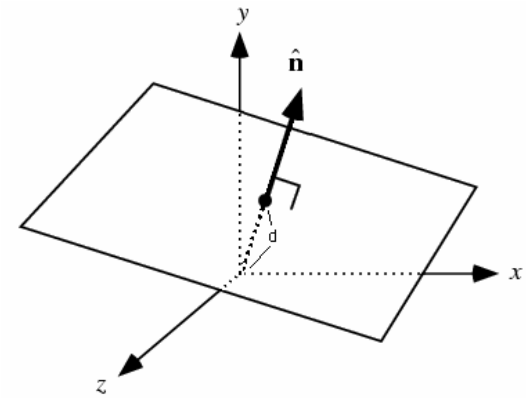
(e) (2 points) Which affine transformation(s) will fix **p, q,** and **r**?

Problem 3: 2D Affine Transformations (cont'd)

(f) (6 points) In general, if M sends **p**, **q**, and **r**, to **p**', **q**', and **r**', respectively, does knowledge of **p**', **q**', and **r**' determine M uniquely? Justify your answer.

Problem 4: 3D Affine Transformations (17 points)

The equation $\hat{\mathbf{n}} \bullet \bar{\mathbf{x}} = d$ describes the plane pictured below which has unit length normal $\hat{\mathbf{n}}$ pointing away from the origin and is a distance *d* from the origin (in the direction of the normal vector).



$$M_{xy} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

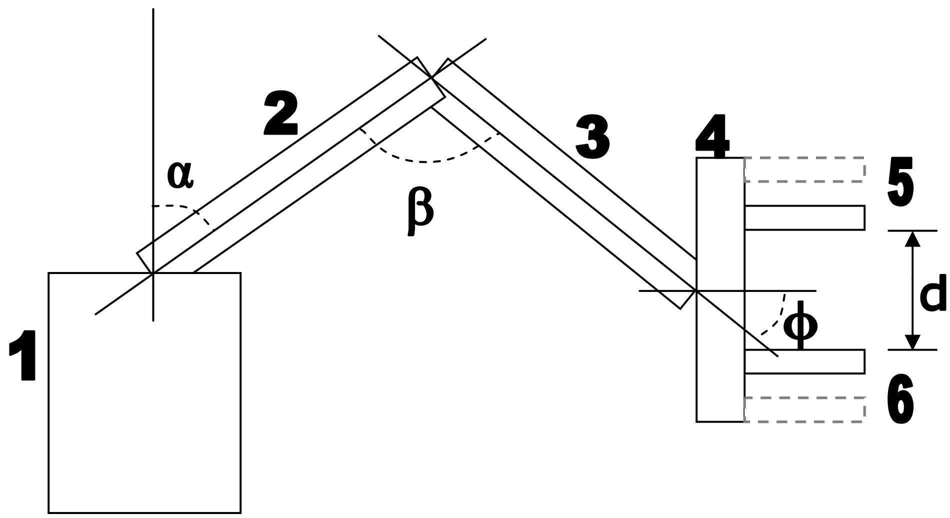
Now consider a plane with normal lying in the *y*-*z* plane. The normal will have the form (0, sinθ, cosθ) for some θ. The equation for the plane is then $y\sin\theta + z\cos\theta = d$. Write out the product of 4x4 matrices that would perform a reflection across this plane. One of these matrices will be a reflection matrix; you must use the matrix M_{xy} above, which performs a reflection across the *x*-*y* plane. You must write out the elements of the matrices, but you do not need to multiply them out. Justify your answer.

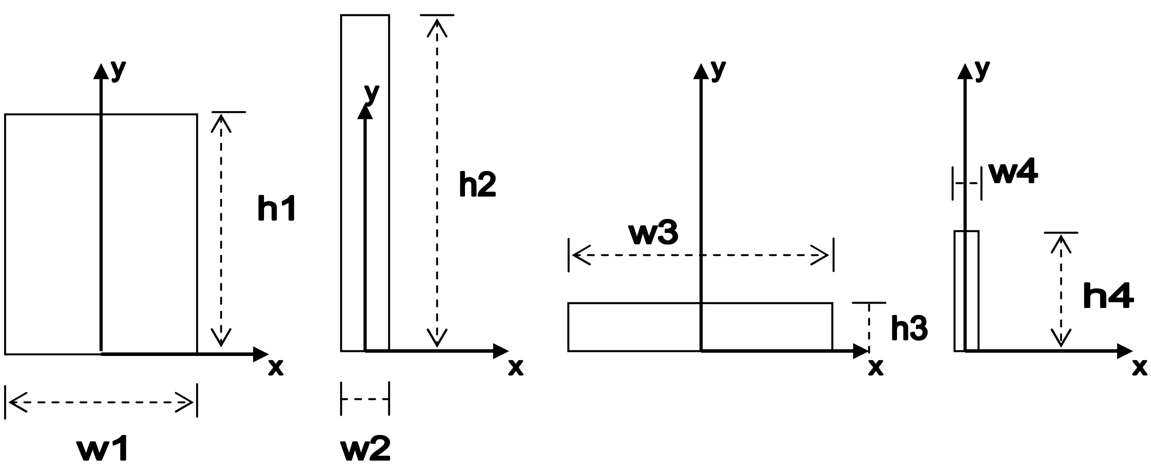
Problem 4: 3D Affine Transformations (cont'd)

(This page is extra space for writing out your answer.)

Problem 5: Hierarchical Modeling (18 points)

Suppose you want to model the robot arm with calipers, shown below. The arm is made out of six parts (**1**-**6**), and each part is drawn as one of the four primitives (**A**-**D**).





A **(Base)** **B** **(Arm Segment)** **C** **(Bar)** **D** **(Caliper)**

The following transformations are also available to you:

- R(θ) – rotate by θ degrees (counter-clockwise)
- T(a, b) – translate by $[a \;\; b]^T$

Problem 5: Hierarchical Modeling (cont'd)

(a) (15 points) Construct a tree to specify the robot arm that is rooted at **1**. Along each of the edges of the tree, write expressions for the transformations that are applied along that edge, using the notation given above (you do not need to write out the matrices). Remember that order is important! The calipers are distance *d* away from each other, and *equidistant from the center of the bar they are attached to*. (Note: Each caliper (objects **5** & **6**) is attached to the bar (object **4**) so you will need to reflect this fact in your tree.)

(b) (3 points) Write out the full transformation expression for the part labeled **5**.