Homework #1

Alpha Compositing, Image Processing,

Affine Transformations, Hierarchical Modeling, Projections

Assigned: Monday, October 20

Due: Monday, November 3
at the beginning of class

Directions: Please provide short written answers to the following questions. Be sure to write your name on the assignment. Please answer the questions on your own and show your work. If you have questions about the assignment, please contact the instructor directly.

Problem 1: Alpha compositing (22 points)

The alpha channel is used to control blending between colors. The most common use of alpha is in "the compositing equation"

$$\mathbf{C} = \alpha\,\mathbf{F} + (1-\alpha)\,\mathbf{B} \quad \text{or} \quad \begin{bmatrix} C_R \\ C_G \\ C_B \end{bmatrix} = \alpha \begin{bmatrix} F_R \\ F_G \\ F_B \end{bmatrix} + (1-\alpha) \begin{bmatrix} B_R \\ B_G \\ B_B \end{bmatrix}$$

where α is the blending coefficient, **F** is the foreground color, **B** is the background color, and **C** is the composite color. In film production, compositing is a common operation for putting a foreground character into a new scene (background). The challenge faced with real imagery is to extract per pixel alpha and foreground color from a live action sequence, to enable compositing over a new background.

(a) (5 points) When filming an actor, a color **C** is observed at each pixel. If the three observed color channel values C_R, C_G, and C_B are the only knowns at a given pixel, how many unknowns remain in the compositing equation at that pixel? Treating each color channel separately, how many equations are there at the pixel? Is it generally possible to solve for all the unknowns under these circumstances? [Note: we are treating each pixel in isolation, so in each of these problems, you should just be thinking in terms of a single pixel.]

(b) (3 points) To assist the process of extracting the desired **F** and α values, the actor may be filmed against a known background, typically solid blue or green. If the components of **B** are known, how many unknowns remain at a given pixel? Is it possible, in general, to solve for **F** and α under these circumstances?

(c) (7 points) When filming the original Star Wars trilogy, the starships were assumed to contain only shades of gray and were filmed against a solid blue background. Thus, at a given pixel, the visual effects people could assume $\mathbf{F} = [L\ L\ L]^T$, where L is a shade of gray, and $\mathbf{B} = [0\ 0\ 1]^T$, where color channel values are in the range [0...1]. Given an observed color $\mathbf{C} = [C_R\ C_G\ C_B\ ]^T$ at a pixel, compute α and L in terms of the color components of **C**. You should comment on how to handle the case when $\alpha = 0$. Show your work. [Note: if the answer is not unique, just provide one possible solution.]

(d) (7 points) Suppose you had the luxury of two consecutive images of a stationary foreground subject against a blue and a green background in succession, $\mathbf{B} = [0\ 0\ 1]^T$ and $\mathbf{G} = [0\ 1\ 0]^T$, thus recording two colors, **C** and **D**, respectively, at each pixel. You would then have to consider two color compositing equations $\mathbf{C} = \alpha\,\mathbf{F} + (1-\alpha)\,\mathbf{B}$ and $\mathbf{D} = \alpha\,\mathbf{F} + (1-\alpha)\,\mathbf{G}$. Solve for α and the components of the foreground color, F_R, F_G, and F_B at a given pixel in terms of the components of **C** and **D**. Show your work. [Note: if the answer is not unique, just provide one possible solution.]

Problem 2: Image Processing (16 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 effect will this normalized filter have when applied to an image?

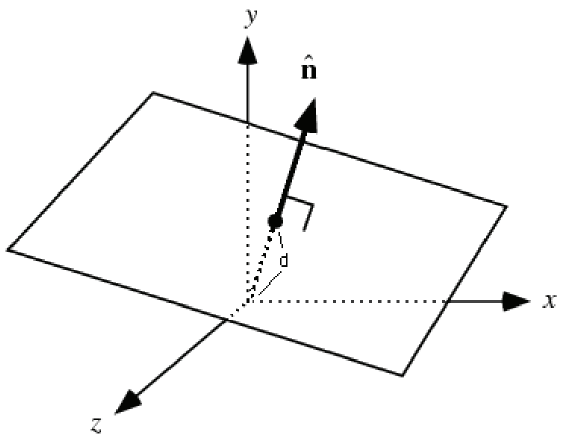
c) (4 points) Compute *A*B*, using *A* and *B* from the ***original*** problem statement, i.e., ***without*** using the scale factors described in (a) and (b). You can treat *B* as the filter kernel and assume that *A* is zero outside of its support. You do *not* need to show your work. [Aside: convolution 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) (4 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? [Answer both of these questions.]

Problem 3: 3D Affine Transformations (20 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).

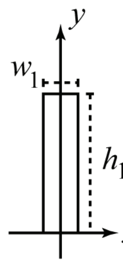


$$M_{xy} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

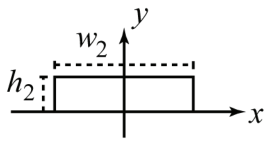
Now consider a plane with normal lying in the *y*-*z* plane. The normal will have the form (0, sinθ, cosθ) for some θ. The equation for the plane is then $y\sin\theta + z\cos\theta = d$. Write out the product of 4x4 matrices that would perform a reflection across this plane. One of these matrices will 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 and the product order in which they would be applied, but you do not need to multiply them out. Justify your answer with words and/or drawings.

Problem 4: Hierarchical modeling (26 points)

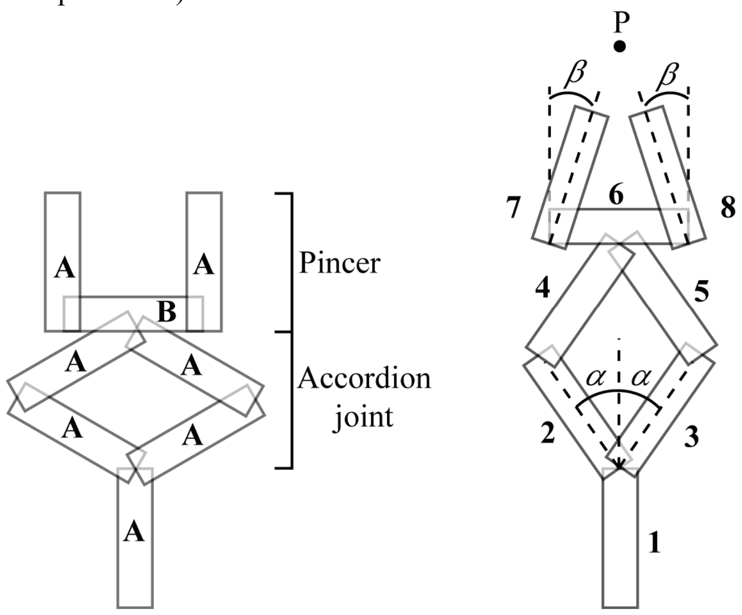
Suppose you want to model the pincer with accordion joint illustrated below. The model is comprised of 8 parts, using primitives **A** and **B**. The model is shown in two poses below, with the controlling parameters of the model illustrated on the far right. The illustration on the right also shows a point P that the model is reaching toward, as described in sub-problem **c**).



Primitive **A**



Primitive **B**



Assume that α and β can take values in the range [0, 90°]. Also assume that all parts use primitive **A**, except for part **6**, which uses primitive **B**. The model on the left shows the primitives used, the model on the right shows the enumeration (naming) of the parts.

The following transformations are available to you:

- R(θ) – rotate by θ degrees (counter clockwise)
- T(a, b) – translate by $\begin{bmatrix} a \\ b \end{bmatrix}$

a) (16 points) Construct a tree to describe this hierarchical model using part **1** as the root. 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 the order of transformations is important! Show your work wherever the transformations are not "obvious." Your tree should conatin a bunch of boxes (or circles) each containing one part number (1…8); these boxes should be connected by line segments, each labeled with a corresponding transformation that connects child to parent.

b) (3 points) Write out the full transformation expression for part **7**.

c) (7 points) Suppose the primitives are infinitesimally thin, $w_1 = h_2 = 0$, and have lengths $h_1 = 10$ and $w_2 = 12$. Assume that part **1** sits right on the origin in world coordinates. What would the α and β parameters have to be so that the model extends out and closes the pincer just enough to precisely grasp the point $P = [0 \quad 28]^T$, in world coordinates. Show your work.

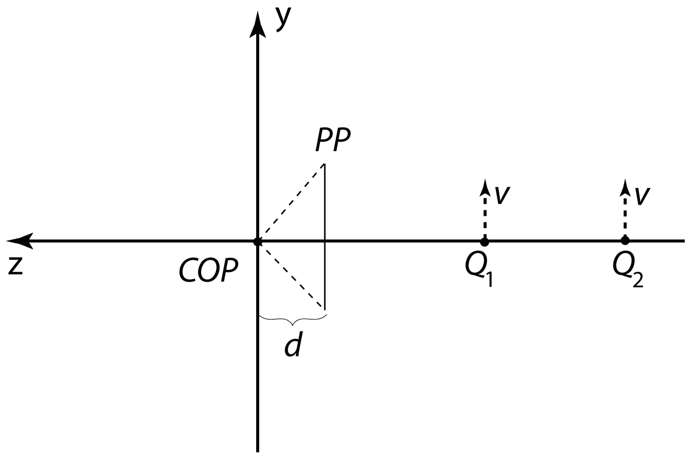
Problem 5: Projections (16 points)

The apparent motion of objects in a scene can be a strong cue for determining how far away they are. In this problem, we will consider the projected motion of points and line segments and their apparent velocities as a function of initial depths.

a) (6 points) Consider the projections of two points, Q_1 and Q_2, on the projection plane *PP*, shown below. Q_1 and Q_2 are described in the equations below. They are moving parallel to the projection plane, in the positive *y*-direction with speed *v*.

$$Q_1(t) = \begin{bmatrix} 0 \\ vt \\ z_1 \\ 1 \end{bmatrix} \qquad Q_2(t) = \begin{bmatrix} 0 \\ vt \\ z_2 \\ 1 \end{bmatrix}$$

$$0 > z_1 > z_2$$



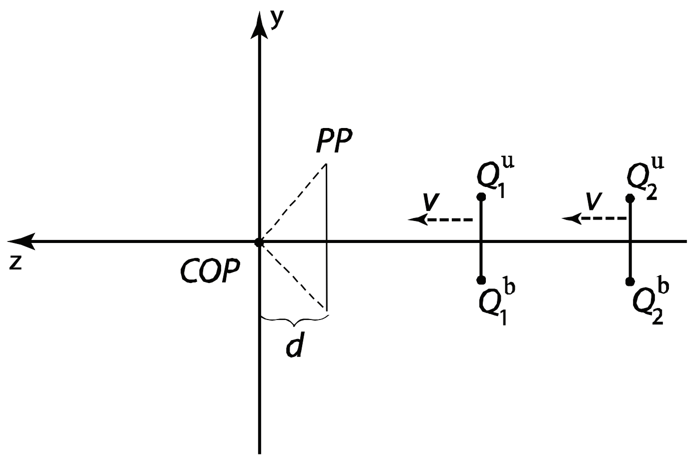
Compute the projections q_1 and q_2 of points Q_1 and Q_2, respectively. Then, compute the velocities, dq_1 / dt and dq_2 / dt, of each projected point in the image plane. Which appears to move faster? Show your work.

b) (10 points) Consider the projections of two vertical line segments, S_1 and S_2, on the projection plane *PP*, shown below. S_1 has endpoints, Q_1^u and Q_1^b. S_2 has endpoints, Q_2^u and Q_2^b. The line segments are moving perpendicular to the projection plane in the positive *z*-direction with speed *v*.

$$Q_1^u(t) = \begin{bmatrix} 0 \\ 1 \\ z_1 + vt \\ 1 \end{bmatrix} \qquad Q_2^u(t) = \begin{bmatrix} 0 \\ 1 \\ z_2 + vt \\ 1 \end{bmatrix}$$

$$Q_1^b(t) = \begin{bmatrix} 0 \\ -1 \\ z_1 + vt \\ 1 \end{bmatrix} \qquad Q_2^b(t) = \begin{bmatrix} 0 \\ -1 \\ z_2 + vt \\ 1 \end{bmatrix}$$

$$0 > z_1 > z_2$$



Compute the projected lengths, l_1 and l_2, of the line segments. Then, compute the rates of change, dl_1 / dt and dl_2 / dt, of these projected lengths. Are they growing or shrinking? Which projected line segment is changing length faster? Show your work.