Review for Final
CMSC 426 – Fall 2005

General comments

There are six key technical ideas in this class. The goal of the class is for you to master these ideas and to see how they can be used to solve problems in vision. So hopefully the final exam will test both your understanding of basic problems in vision and your mastery of these techniques, and how they can be used to solve vision problems. Below is an outline of the class. I'm including some sample problems that you can work if you want practice on some of these problems. The practice problems from the midterm are also appropriate.

Correlation/Convolution

The first key idea of the course is to understand correlation and convolution.

1. You should understand how to perform these operations.
2. You should also understand some of their basic properties. Convolution is associative, and this will allow you to combine operations, or sometimes to split operations. For example, we can combine smoothing and a derivative filter into one operation. Or we can break smoothing with a box filter into two convolutions with smaller, 1D filters.
3. Finally, you should know how to construct a filter so that you can use convolution and correlation to perform a few different operations, including smoothing an image, taking a derivative, finding a region in an image that provides a good match to a small template, or matching a template of edges to a set of image edges, using the distance transform and Chamfer matching.
4. * The Fourier series representation of functions might figure in a challenge problem.

SAMPLE PROBLEMS

1) Compute the convolution of the 1D image: (0,0,1,1,2,2,3,3,4,4,4,4) with the convolution kernel: (1/4,1/2,1/4). Treat the boundary in any reasonable way.

2) Compute the distance transform of the following image (1 means an edge is present):

0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	0

Explain how to use this to find the best match to the model:

1	1	0	0
0	1	0	0
0	1	0	0
0	1	0	0

3) Suppose I convolve an image first with the filter [1; 0; -1] and then with the filter [-1 0 1]. Give a single, 2D filter that will accomplish the same thing. What is the mathematical operation this filter performs?

The Image Gradient

The image gradient is perhaps the most fundamental way we have of representing images. It captures how the image changes. You should understand:

1. How to compute the gradient.
2. Given a gradient, find the magnitude and direction of the gradient.
3. Use the gradient to find boundaries. This includes understanding whether a gradient is a local maximum in the direction perpendicular to the gradient.
4. Use the gradient to find corners, based on whether a region has strong gradients with multiple directions.
5. Use the gradient and the temporal derivative of an image to compute the optical flow.

SAMPLE PROBLEMS

1) Consider the following 1D image:
(0 0 …. 0 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 0 0 0 10 10 10 0 0 0 0 …)
Where would you expect a 1D edge detector to find edges?
An edge detector has a parameter that allows it to ignore weak edges. As we increase this parameter, which edges disappear first?
If we smooth the image before detecting edges, how would you expect this to affect the position and number of the edges?

2) For the point 16, in bold face, find the gradient. Give the direction and magnitude of the gradient.

5	8	11	14	17	20
6	9	12	15	18	21
7	10	13	**16**	19	22
8	11	14	17	20	23
9	12	15	18	21	24
10	13	16	19	22	25

4) Give an example of a 6x6 image region for which corner detection will declare a corner to be present in the center pixel, because the corner matrix has two large singular values. Give an example of a region which will

produce a corner matrix with only one large singular value. Give a region that produces small singular values.

5) Play with the corner detector you implemented for PS 7. Try to find a 6x6 region that will produce the strongest corner.
6) I is an image described by the equation I(x,y) = x*x + y*y. J is the same image translated 1 pixel in the positive x direction. Suppose I and J are consecutive images in a motion sequence. Write the optical flow equation in its general form. Using this equation, give the equation for a line in the second image on which we expect to find the point that was at (5,5) in the first image. Perform the same computation for a point at (5,0). Combine these two equations, and show that this yields the translation of the image.

Probabilistic Modeling

This is a complex topic that we have only touched on. There are two key parts to probabilistic modeling. First, you must use some prior knowledge to construct a statistical model. Second, you must use this model, typically to classify new data or to generate new examples.

1. Building statistical models
    a. Given some independent samples, use a histogram to build a distribution for them.
    b. Given some samples, construct a Markov model of the data. While there are more sophisticated ways of doing this, we have discussed the approach of using previous examples for your Markov model.
2. Show how to use each type of model to either classify examples or to generate new examples.
3. Applications: we have studied these problems with the examples of background subtraction and texture synthesis.

SAMPLE PROBLEMS

1 Suppose we are performing background subtraction. For one particular pixel, we see the following intensities: 1 1 2 2 3 3 3 3 1 1 2 2 3 3 3 3 1 1…. If we use a histogram of these values and no Gaussian smoothing (that is, kernel density estimation with a sigma of 0) what would you estimate is the probability that the next pixel would have an intensity of 2?

2 Suppose we use a Markov model, and estimate the probability of a pixel based on the intensity of the previous pixel. If the last pixel had an intensity of 2, what would you estimate is the probability that the next pixel will have an intensity of 2? Of 3? Of 1?

3 Consider the following statement: “When performing background subtraction with kernel density estimation, we should use a smaller and smaller value for sigma as the size of our training set increases.” Do you think this is true? Why or why not?

Optimization

Optimization is a very large area. There is a wide array of optimization methods that have been applied to vision problems. You should understand the few of these methods that we have studied, and how they can be used to solve vision problems.

1. Structure of optimization
    a. Define a set of possible solutions.
    b. Define a cost function that defines the value of each solution.
    c. Find a search algorithm for exploring solutions, picking the best one found.
2. Optimization methods
    a. Brute-force search. For many problems, at least as a baseline, the starting point is to try every possible solution.
    b. Shortest path algorithm.
    c. K-means algorithm.
    d. Multi-scale
    e. RANSAC
3. Applications
    a. Finding boundaries.
    b. Clustering pixels.
    c. Stereo matching
    d. Template matching

SAMPLE PROBLEMS

1. Kmeans: Suppose we have points at the locations: (1,3) (4,7) (2,9) (2,2) (3,6), and we pick centers at (1,4) (4,4). Which points will be assigned to each center? What will be the location of the new centers?

Suppose we have five points and two centers, as above. Place an upper bound on the number of possible iterations that k-means can perform before it converges.

2. Suppose we are performing stereo matching, and we want to add a penalty, D, which we have to pay whenever there are any changes in disparity. Explain how to build a graph so that we can encode this penalty, and still use a shortest path algorithm to find the best match.

3D Geometry

This all comes down to forming linear subspaces and intersecting them. For example, given two points, find the line that they form. Or given three points, find the plane that they form. Or intersect a line with a plane, or two planes to find a line. With these tools, we can solve a variety of problems:

1. Perspective Projection: form a line from two points and intersect it with an image plane.
2. Locating a 3D point from its appearance in a 2D image: form a line from two points and intersect it with any other possible constraint.
3. Locating a 3D point from its appearance in two 2D images (stereo): form a line from two points twice, and intersect these lines. For a standard stereo set-up, this can be solved more simply using similar triangles.
4. Epipolar geometry:
    a. Given a point in a 2D image, delimit its possible location in a second image (the epipolar constraint): form a plane with three points (two focal points and an image point) and intersect it with an image plane.
    b. You should also understand how to find the epipole, by forming a line between the two focal points and intersecting this with an image plane.
5. Image rectification: project an image plane onto a new image plane.

SAMPLE PROBLEMS

1) If using perspective projection, how can you tell whether a line in the world will project to a single point in the image?

2) Consider a camera in the standard position for perspective, with the image plane the z=1 plane, and the focal point at the origin. Consider a rectangle with corners at: (0,0,2), (1,1,2), (0,0,3), (1,1,3). Give its image with perspective projection. Prove that parallel lines in the world don't necessarily project to parallel image lines.

3) Suppose we have two parallel lines. One goes through (0,0,2) to (0,0,3) and the other through (1,0,2) to (1,0,3). What will be their vanishing point?

4) Suppose we have a camera with a focal point at (7,2,5) and an image plane of z = 10. We see a point at (12,9,10). What is the equation for a line in the world that contains this point? Suppose we have another camera with a focal point of (17, 3, 2), and an image plane of z = 6. What is the epipolar line in this image plane where the image of the point must appear? Pick a point that is on this line. If we see the point appear there, what is its 3D position?

Motion and pose using Linear Algebra

The key ideas we have discussed concern the representation of motion and projection using matrices. To use this, you should know how to:

1. Represent a (2D or 3D) point as a vector.
2. Represent a (2D or 3D) translation using a matrix.
3. Represent scaling using a matrix.
4. Rotations are a bit more complicated. To be more explicit, you should know how to:
    a. Represent a rotation with a matrix.

    b. Determine whether a matrix represents a rotation.
    c. Represent rotation about a point other than the origin using a matrix.
5. Represent scaled orthographic projection using a matrix. This scales the *x* and *y* coordinates of a point, while removing the *z* coordinate.
6. Now you can represent projection + motion with an equation like I = TS. You should understand how to solve this equation, given parts of it. For example, given I and the first four points in S, find T.
7. You should also understand how to use this for pose determination, for use in RANSAC.

SAMPLE PROBLEMS

1) Suppose there is a 2D rotation, translation, and scaling that moves the point (1,0) to (3,3), and that also moves the point (2,0) to (4,4). What is the matrix that encodes this transformation?

2) Consider a 2D model consisting of three points, with x,y coordinates given by the columns of:

0	20	10
0	0	10

and an image consisting of the points:

4	22	12	16	10	3
0	1	12	19	18	20

Simulate one step of RANSAC with bounded error of 3 pixels to recognize the model, allowing for 2D translation, rotation, and scaling. That is, match enough features of the model and image to determine 2D rotation, translation, and scaling. Project the remaining model features into the image using this transformation. Where do they appear? Look for matching image features within a distance of 3 pixels. Do you find them? For this example, how many different poses can be generated by matching model features to image features? How many involve correct matches between model points and their corresponding image points, assuming that each model point appears in the image.

3) Start with points, P, that are the corners of a cube. Construct a motion matrix, S, that will rotate these points by pi/8 about the x axis and then pi/4 about the y axis, translate the points by (1, -1), and project them into the image, scaling by 1/2 (use matlab to help if you want). Compute I = S*P.

4) Let I be a matrix in which each column gives the x,y coordinates of a point in two images.
I =

3	1	1	1	3
1	3	0	2	2
4	4	1	3	5
2	2	2	2	1

and let P be a matrix in which each column gives the x,y,z coordinates of a 3D point that generated the matching column in I.

P =

3	1	1	1	3
1	3	0	2	2
2	2	2	2	1

compute the motion matrix, S, that could generate this image.