

CS4495/7495 Computer Vision “Mid-Term” Sample Solutions

Answer each of the 4 questions. They are worth 25 points each. Show your work, and mostly show your thoughts. If you can't do a problem completely, do it as well as you can.

1. (10 pt) This concerns the motion equations of Adiv's paper and that we did in class. Suppose some point P in the world is moving with velocity v where

$$v = -\dot{t} - \omega \times r$$

where $r = \langle X, Y, Z \rangle$ is the vector from the camera origin to the point P , $t = \langle A, B, C \rangle$ is the translation vector, and $\omega = \langle U, V, W \rangle$ is the rotation vector.

- (a) Remembering that x and y in the image are X/Z and Y/Z respectively (Assuming focal length f is one), show that:

$$\dot{x} = -Uxy + V(1 + x^2) - Wy + (A - Cx)/Z$$

$$\dot{y} = -U(1 + y^2) + Vxy + Wx + (B - Cy)/Z$$

Solution

$$w \times r = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ U & V & W \\ X & Y & Z \end{vmatrix} = (VZ - WY)\vec{i} + (WX - UZ)\vec{j} + (UY - VX)\vec{k}$$

$$v = -\dot{t} - \omega \times r = \begin{bmatrix} -A - VZ + WY \\ -B - WX + UZ \\ -C - UY + VX \end{bmatrix} = \begin{bmatrix} -\dot{X} \\ -\dot{Y} \\ -\dot{Z} \end{bmatrix}$$

$$\begin{aligned} \dot{x} &= \frac{\dot{X}Z - X\dot{Z}}{Z^2} = \frac{\dot{X}}{Z} - \frac{X}{Z} \frac{\dot{Z}}{Z} = \frac{\dot{X}}{Z} - x \frac{\dot{Z}}{Z} \\ &= \frac{A}{Z} + V - W \frac{Y}{Z} - x \frac{(C + UY - VX)}{Z} \\ &= \frac{A}{Z} + V - Wy - \frac{C}{Z}x - Uxy + Vx^2 \\ &= -Uxy + V(1 + x^2) - Wy + (A - Cx)/Z \end{aligned}$$

$$\begin{aligned} \dot{y} &= \frac{\dot{Y}Z - Y\dot{Z}}{Z^2} = \frac{\dot{Y}}{Z} - \frac{Y}{Z} \frac{\dot{Z}}{Z} = \frac{\dot{Y}}{Z} - y \frac{\dot{Z}}{Z} \\ &= \frac{B}{Z} + W \frac{X}{Z} - U - y \frac{(C + UY - VX)}{Z} \\ &= \frac{B}{Z} + Wx - U - \frac{C}{Z}y - Uy^2 + Vxy \\ &= -U(1 + y^2) + Vxy + Wx + (B - Cy)/Z \end{aligned}$$

- (b) Consider the following from an intuitive perspective:
“With a XXXXXX focal length lens it is difficult to tell rotation from translation.” (Think about translation along x vs rotation about y .)
Is XXXXXX ‘long’ or ‘short’; why?
- (c) Now explain your answer to (b) using the derived equations in (a). (Hint: this has something to do with f — the quantity you assumed was 1. What is the effect of f on (a)? If you want, you could re-do (a) using a non-unity f but that’s a lot of work.)

Problems 1 b) and c) are worth a total of **(5 points)**.

Additional **+10 points** were given to everybody for these two problems.

2. **(25 pt)** Morphology question

See additional file on class the web page.

3. (25 pt) You learned about the Hough transform for finding parametric objects like lines or circles or cones. Another approach to structure finding in somewhat noisy images is called **RANSAC** (RANdom SAmple Con-sensus). It works as follows: Suppose you have a collection of edge points in an image and you are trying to find, say, lines. Randomly pick two edge points. This defines your line. Now go along that line in the image and see how many edge pixels you explain. If it's significantly above chance, that's a real line. Mark (i.e. remove) the edge pixels "explained" by that line and continue until not finding any more good lines.

Answer the following: (a) How does RANSAC compare to Hough? Which would work better for, say, finding circles on the Yellow pages images you used? Why? (b) What is a situation you would rather use RANSAC than Hough? Consider the dimensionality/complexity of the two methods?

The issues here include (at least)

- (a) The complexity of the representation versus the number of combinations of image features,
- (b) The ability to define a noise model in the image as opposed to the parameter (accumulator) space.
- (c) The density of "real features" versus noise features in the image.

First, consider the complexity of the two methods. Let us assume there are n feature points (or maybe edges) in the image. For Hough, the time complexity for the voting is *not* just $O(n)$. Rather, it is $O(n b^{(d-1)})$ where b is the number of bins per dimension and d is the dimensionality of the voting space. (Example of line finding: You need to draw a "line" in voting space for each edge point. That line is about b long.) You might assume that b is not too big but $b^{(d-1)}$ can be sizable. Then the "detection" phase is on the order of b^d though that can be sped up if necessary by a hierarchical representation of the space (a la quad-trees). The space complexity is also b^d .

The "voting" complexity for RANSAC is related to ${}_n C_k$ ("n choose k") where k is the number of features needed to define the object (for a circle it's three points or one point and one located and oriented edge). The reason I say related is that one does not need to check all ${}_n C_k$ to find the first set of k that define the given object. In fact, if there are m features on a given object then $m C_k / n C_k$ of the feature sets are from that object. How many sample sets you need to draw from is function of that fraction.

Given the above analysis it is clear that one would want to use RANSAC if you have not much noise but several objects. Also, the more parameters needed to define the model the bigger the voting space gets (exponentially). The number of possible sample sets goes up too but sometimes space is more valuable than time. Also, one can often do better than purely picking features at random to find a set that belongs to the same object.

The issue of the noise model is this: Suppose the object is not quite a circle or the edge point locations are noisy. Then in Hough you get misalignment in the voting space and you need to define an ad hoc scheme in voting space to decide how to blur the votes. In RANSAC you can explicitly define an error model in the image plane and look for features consistent with that error model. Also, you can find all the "close" points and then re-estimate the object parameters and then re-check.

Finally, it should be pretty clear that if there are lots of non-object edges then RANSAC might not work so well. The Yellow Pages task is an example of this. Might need to check too many sample sets to find the object. Having said that, I've seen cases where people use RANSAC quite well even in situations with lots of non-object edges. The trick is to use simple heuristics to improve chance of finding a valid set.

4. In our discussions of edge detection, we discussed a variety of operators. Each was a mask of some sort, typically named after the inventor! We also talked about the Marr and Hildreth operator which looked at the zero crossings of the second derivative of a Gaussian filtering of the image.

- (a) **(12 pt)** Explain why the second derivative of a Gaussian might be a good edge detector? (Draw pictures if you like, and think in 1D if 2D is too hard.)

First, if you Gaussian blur an ideal edge you get a nicely rounded slope up. The 2nd derivative starts out zero, goes positive then back to zero at the inflection point, then negative and then back to zero. Zeros are easy to find and thresholding on the first derivative magnitude makes it easy to mask off the uninteresting ones.

Next, because convolution and differentiation are linear, one can just convolve the image with the second derivative of a Gaussian instead of taking the numerical derivative of the Gaussian blurred image. This is **much** better than doing numerical approximation of a local derivative.

Third, in 2D the second derivative of a Gaussian is the “Laplacian” operator which is a symmetric operator — so no directional bias in detecting edges.

- (b) **(8 pt)** What can you say about the density of edges in the image such an operator might give you? Assume that you threshold the edge finding so that you only accept zero-crossings of sufficient slope.

Because you’ve blurred the image you control how much “high frequency” remains (more technically you dramatically reduce the power of the high frequencies). What high frequency really means is oscillations over a short distance in the image; oscillations are what give edges. Thus by blurring with the Gaussian you can control how dense the edges are: For a given threshold of edge, the bigger the Gaussian the lower the frequency that can have enough power to make it above threshold. Thus the less dense the image edges.

- (c) **(5 pt)** One typically cannot say much about the density of edges many ad-hoc operators might give you (like Sobel). Why can you say more about the edges of something designed like Marr-Hildreth than say Sobel or Prewitt?

The Marr-Hildreth operator is based upon real image processing principles and therefore one can do frequency analysis. In particular, because convolution with a derivative of a Gaussian function is equivalent to multiplication with a different analytic function in the frequency domain, you can model precisely what will happen.

If you don’t know about frequency analysis you should still be able to realize that blurring allows control over high frequency power, and that the linearity of the operations permits convolving with an analytic function so that you can state precisely what will happen to an image.