

CPS296.1- Homework 4

Due on October 31, 2007

Questions may continue on the back. Please write clearly. What I cannot read, I will not grade.

1. Rodrigues rotation vectors are obviously not additive: the rotation with Rodrigues vector $\mathbf{r} = \mathbf{r}_1 + \mathbf{r}_2$ (even if this vector remained within the ball of all rotations) is generally not the composition of the two rotations with Rodrigues vectors \mathbf{r}_1 and \mathbf{r}_2 . There is, however, one set of exceptions. What relationship must hold between \mathbf{r}_1 and \mathbf{r}_2 for their sum $\mathbf{r} = \mathbf{r}_1 + \mathbf{r}_2$ to represent the composition of the two rotations corresponding to \mathbf{r}_1 and \mathbf{r}_2 ?

2. This problem explores bias in a simple estimation problem. The function $f(x, a, b)$ available on the homework page is defined as follows:

$$y = f(x, a, b) = e^{-\frac{1}{2}\left(\frac{x-a}{b}\right)^2}. \quad (1)$$

Other than for a scaling factor, this function has the shape of a Gaussian density function with mean a and standard deviation b . However, we merely regard this as a function we want to fit to given data. To obtain the data for this assignment, type

$$[x, y] = \text{data}(n, a, b, u);$$

where the function `data` is also provided on the homework page. The arguments a and b have the meaning explained earlier. Set $a = 0$ and $b = 1$ in the remainder of this problem. The argument n is the number of data points. Data values in y are corrupted by uniform, zero-mean noise in the interval $[-u/2, u/2]$, and the fourth parameter u specifies the width of this noise interval.

(a) Equation (1) can be rewritten by taking the logarithm of both sides and multiplying by -2 :

$$-2 \log y = \left(\frac{x-a}{b}\right)^2 = c_1 x^2 + c_2 x + c_3$$

with suitable expressions for the coefficients c_1, c_2, c_3 . These coefficients appear linearly, so they can be estimated from the data as a linear fitting problem. Write a Matlab function

$$\text{function}[a, b] = \text{polyEstimate}(x, y)$$

that estimates a and b using the Matlab function `polyfit` and the transformation above. Make sure you do not get complex results (as opposed to real) by using only the appropriate data samples to compute your estimates (the logarithm of a negative number is complex).

(b) Run your function on data generated by `data` with n set to 10, then 100, then 1000. Show a single diagram with four curves on it: The functions $f(x, \hat{a}, \hat{b})$ that use your estimates, as well as the true function $f(x, a, b)$. Plot all these for

$$x = \text{linspace}(x_{\min}, x_{\max}, 100);$$

where x_{\min} and x_{\max} are the smallest and largest values of x you encounter in your entire run. Distinguish your plots by color or line style and add a legend to your plot (`help legend`) so it is clear which plot is which.

(c) Briefly summarize your observations about your plot: which estimates are good, which are not, and in what way.

(d) For $n = 1000$, plot the function $c_1 x^2 + c_2 x + c_3$ as a connected plot (and for a regularly spaced vector x of points), and superimpose the cloud of points $(x_i, -2 \log y(x_i))$ on the same graph. Use this plot to explain qualitatively the cause of the main problem(s) you encountered in your previous answer.

(e) A single experiment does not tell much about either bias or variance. Run your experiment again for $n = 20, 50, 100, 200, 500, 1000$. For each value of n , run 100 trials (calling the function `data` anew for each trial, so you obtain different random samples). Use the Matlab `errorbar` function to create a plot of the average values (over the 100 trials) of a for each value of n , with error bars obtained from the standard deviations (over the 100 trials) of a .

(f) Repeat for b .

(g) Write a function

$$\text{function}[a, b] = \text{refineEstimate}(x, y, a_0, b_0)$$

that uses the Matlab function `fminsearch` with a termination tolerance `TolX` (find out what this means by typing `help optimset`) of 10^{-6} to refine the estimates from `polyEstimate`. Hand in your code.

- (h) On new diagrams, or, preferably, superimposed on the previous plots, show the error-bar diagrams for the refined estimates. Briefly comment on bias and variance in your results, for both parameters a and b .

3. Write a Matlab function

```
function F = fundamental(x1,x2)
```

that computes the fundamental matrix F from the coordinates of corresponding points in two images. The matrices $x1$ and $x2$ have size $n \times 2$. Normalize F so that its element of greatest magnitude has magnitude 1.

- (a) Hand in your code.

(b) Write a Matlab function

```
function drawEpiLines(xvec,F,fig,x)
```

that draws epipolar lines in one image. The figure number `fig` is assumed to already have the image displayed. Remember to set `hold` to `on` so that the image is not erased when you plot over it. The $n \times 2$ input `xvec` is either `x1` or `x2` as above, and F is the fundamental matrix. The argument `x` has size 1×2 and contains the index of the first and last column in the image (that is, the smallest and greatest x coordinate of any image point). Hand in your code.

- (c) Use the Matlab function `ginput` to record in `x1` and `x2` the coordinates of corresponding points in the two images `left.JPG` and `right.JPG` available on the homework web page. Run your function `fundamental`, and hand in the resulting fundamental matrix F . Four decimal digits are enough.

- (d) Run your function `drawEpiLines` twice to draw the epipolar lines in the two images. Also draw *large* (use property `'MarkerSize'`, 12 or greater in `plot`) dots corresponding to the image points in `x1` and `x2`. Check that the lines go approximately through the dots. They may not go through them exactly because of lens distortion and possible errors in determining the points. Hand in images with lines and dots superimposed.

4. Write a brief proposal for the project you plan to work on for this class in the remainder of this semester. The proposal is due later than this assignment, and should be mailed to me on October 31 in a separate PDF. **You need not hand in anything for this proposal on the due date of this homework assignment**, but you should start to work on this proposal now.

Your proposal should be about one page long, and contain the following items:

- Your name.
- A simple, short, informative title. Example: *A Fast Implementation of a Block Stereo Matcher*.
- A brief but self-contained description of what you plan to do. Example:

A *stereo matcher* takes as its input two images taken by cameras in the standard stereo configuration, as well as a maximum disparity value d_{\max} . For each pixel at (x, y) in the left image, the matcher computes a *disparity*, that is, a value d in the range $0 \leq d \leq d_{\max}$ such that pixel $(x - d, y)$ in the right image is likely to correspond to pixel (x, y) in the left.

More specifically, a *block stereo matcher* computes disparity for left pixel (x, y) by comparing a fixed-size block of pixels centered at (x, y) with blocks in the right image and centered at $(x - d, y)$ for all integer values in the allowed range for d . The matcher returns the disparity d that yields the best comparison, that is, the right-image block that is most similar to the one in the left image.

I plan to implement a block stereo matcher based on the measure of Sum of Absolute Differences (SAD):

$$SAD(x, y, d) = \sum_{a=-h}^h \sum_{b=-h}^h |L(x, y) - R(x - d, y)|$$

where L and R are the left and right image (assumed to be black-and-white), and h is half the block size.

A straightforward implementation of this definition results in an algorithm with a running time that is $O(nh^2d_{\max})$ where n is the number of pixels in the each image. However, many of the same operations are repeated when computing $SAD(x, y, d)$ for the same value of d and nearby values of x or y . By storing running sums, it was shown [*insert proper citation here*] how the running time can be made independent of the size of the block.

I will implement both the slow and the fast version in the C programming language and test them on images from the CMU Computer Vision Web Page, <http://www.cs.cmu.edu/~cil/vision.html>. In my study, I will focus in particular on running times. However, I will also take this opportunity to comment on strengths and weaknesses of a block stereo matcher.

- Well formatted references to the literature you intend to read. A small number of (up to three or four, but perhaps even fewer) articles, web pages, and/or book sections will suffice.
- A description of the form in which you will present your results. In any case, your final presentation will include a report with a description of your code and experiments. In addition, you may choose to make your code available on the web, either as source or as an applet, or to put your final discussion on the web.
- A schedule of work, including how much time you plan to spend on each stage of your work. This is not a commitment, but a plan. The schedule can be at a coarse level of granularity: one week for a , ten days for b , and so forth.
- A discussion of which parts will be most challenging, what might go wrong, and what to do in case of failure. Keep in mind that the goal of the project is to help you learn some aspect of computer vision, not to invent something new. For instance, negative results are fine, as long as they are analyzed well.