CS 498/698: Assignment 2, Robust Estimation of Image Features
Due: 1pm, Thurs. Feb. 24, 2000.

Many image features, such as image lines, curves, local image velocity, and local stereo disparity, can be expressed in terms of parametrized models. In order to estimate the model parameters, say given by $\vec{a} \in \Re^d$, we can often derive *linear* constraints on these parameters from image measurements. In particular, suppose various combination of image measurements provide a set of constraint vectors $\{\vec{c}_k\}_{k=1}^{K}$. And assume that each of these observed vectors, $\vec{c}_k$, provide constraints on the image feature parameters, $\vec{a}$, through an equation of the form

$$\vec{c}_k \cdot (\vec{a}, 1) = 0. \tag{1}$$

In this course we consider various instances of this situation, including the estimation of parameterized curves, image motion, and 3D motion.

One serious complication is that we are rarely sure that all of the constraint vectors $\vec{c}_k$ are appropriate for any *single* parameterized model. For example, when estimating image lines, individual edgels give rise to these linear constraints. But there are typically multiple lines in the image, along with curves, and also extraneous edgels. The constraints provided by these other lines, curves and extraneous edgels must be ignored when estimating the parameters for any particular line. Therefore, in order to estimate the parameters for one line, we must somehow use just the subset of the constraint vectors that apply to that line. But then how do we decide which subset (or subsets) to use? A similar situation arises in the estimation of image motion, where several different image velocities may occur in an image region due to the presence of several objects and/or occlusion.

In this assignment, we will use the terminology of image motion, where the parameters $\vec{a} \in \Re^2$, represent the horizontal and vertical components of image velocity. This gives a relatively simple 2D estimation problem for $\vec{a}$. However, virtually any other parameterized model could be used here instead.

Our objective here is to investigate several approaches for solving problems of this type, and to identify some of the trade-offs and design issues in getting the chosen algorithms to run effectively.

In particular, we will consider the optimization problem for $\vec{a}$, where we seek a minimum of the error function

$$E(\vec{a}) = \sum_{k=1}^{K} \rho(\vec{c}_k \cdot (\vec{a}, 1)). \tag{2}$$

Here $\rho(e)$ is the "estimator." You will investigate different choices for ρ, and different ways to generate an initial guess.

a) **Least Squares.** Write a Matlab or C program to minimize (2) for the special case of $\rho(e) = e^2$. Your program should use the `randomConstraints` program (provided in the gzipped tar file `randomConstraints.tar.gz` on the course homepage) to generate sets of constraint vectors $\{\vec{c}_k\}$. This procedure generates test cases involving constraints from one or more motions. Use `sigMeas` = 0.25 as the standard deviation of the measurement noise during the generation of the constraints. Show the results from the least squares approach when the data set consists of one and two distinct motions.

b) **Robust M-Estimator**. Modify your program in part (a) to use the estimator

$$\rho(e; \sigma) = \frac{e^2}{\sigma^2 + e^2}. \tag{3}$$

Here σ is the scale parameter for the estimator. Note that for errors e significantly larger than σ the cost, $\rho(e; \sigma)$, is roughly constant, unlike the squared error used in part (a). Your program will need to use an iterative scheme to solve the minimization problem for the error function given in (2). In class we described an iterative approach based on successive application of a weighted least squares estimator. You may use this method, or any other optimization technique you like (eg., gradient descent with a variable step size is a viable alternative). Use various values of σ between 0.2 and 2, and try your program on the same test cases you used for (a). Also try various initial guesses. Comment on the quality of your results, and discuss the behaviour of your program for various values of σ.

c) **Deterministic Annealing.** Here we consider an approach for making your procedure in (b) less sensitive on the initial guess, especially for small values of σ.

Modify your program in part (b) to solve a sequence of minimization problems, which begin with a large value of σ and slowly reduce it. In particular, first use the quadratic estimator in (a) to provide an initial guess for the velocity. Next use your procedure in part (b) to solve the minimization problem for a large value of σ (say 2.0). Then reduce sigma by a factor of κ (try values of κ between 0.5 and 0.95), and restart the robust minimization in part (b) again using your latest result as the initial guess. Repeat this "deterministic annealing" of σ until a sufficiently small value is reached (say 0.2). Comment on the behaviour of this approach.

In order to estimate a second motion, try the following. Given a solution $\vec{a}_0$ provided by the previous deterministic annealing routine, delete all constraints with a small value for the estimator $\rho(\vec{c}_k \cdot (\vec{a}_0, 1); \sigma)$ (say, corresponding to an error within 2.5σ). Then reapply the above routine to the remaining constraints. This process can be terminated when the number of remaining constraints becomes too small. Discuss how to decide what 'too small' should be (there is no good answer here).

d) **Random Sampling**. An alternative method for getting a useful initial guess is to randomly select pairs of constraints, say $\vec{c}_i$ and $\vec{c}_j$, for which the associated constraints on $\vec{a}$ are in equation (1) are linearly independent. From each such pair, construct the parameter vector $\vec{a}_{i,j}$ which satisfies both equations $\vec{c}_i \cdot (\vec{a}, 1) = 0$ and $\vec{c}_j \cdot (\vec{a}, 1) = 0$. Given this $\vec{a}_{i,j}$ construct a histogram (i.e. the data for a bar graph of frequency of occurrence versus error) of the squared residuals

$$((\vec{a}_{i,j}, 1) \cdot \vec{c}_k)^2, \tag{4}$$

for $k = 1, \ldots, K$ (alternatively, you can use the estimator (3) with a large value of σ). Finally, from this histogram, estimate the median residual, $Med(\vec{a}_{i,j})$, over the data set $\{\vec{c}_k\}_{k=1}^{K}$. (The use of the histogram avoids the need to sort the residuals.)

Do this whole sampling procedure for m independent samples of constraint pairs $\vec{c}_i$ and $\vec{c}_j$. From these samples, select the particular $\vec{a}_{i,j}$ which gave you the minimum median residual, $Med(\vec{a}_{i,j})$. Then start your procedure in part (c) with the initial guess, $\vec{a}$, set to be this selected value of $\vec{a}_{i,j}$. Also, for the initial value of σ, use a value estimated from the median residual.

In order to select m, the number of pairs to use, consider the following problem. Consider sampling from a Bernoulli distribution in which a fraction p of the constraints are from a particular motion. Suppose the remaining constraints (with probability $1 - p$) are from outliers. Suppose $p \geq 0.5$. How many random pairs do you need to use to have a 95% chance of getting at least one pair from the particular motion?

The procedures above depend on a large number of tunable parameters. I do not expect extensive testing of the methods over all these parameters. Instead, I will be looking to see if you understand the main issues. In particular, I expect you to identify some of the trade-offs and some of the design issues in getting the algorithms to run effectively.