CS787: Assignment 2, Robust and Mixture Models for Optic Flow
Due: 5pm, Wed. Nov. 8, 2004.

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.

Constraint Lines for Optical Flow

We write the *Brightness Constancy Condition* (BCC) as:

$$I_x v_x + I_y v_y + I_t = 0 \tag{2}$$

where $\vec{a} = (v_x, v_y)$ is the motion and the derivative information observed at each pixel is $\vec{c}_k = (c_{kx}, c_{ky}, c_{kt})$, with $c_{kx} = I_x$, $c_{ky} = I_y$, and $c_{kt} = I_t$. For each measurement we get one contraint of the form $\vec{c}_k \cdot (\vec{a}, 1) = 0$.

Synthetic Flow Data

I generated several synthetic datasets to test the algorithms. Each dataset contains one, two, or three motions. Datapoints are generated by selecting one of the motions (according to their mixing proportions) adding noise (standard deviation $\sigma_{meas} = 0.25$) and outputing the constraint line. The datafile contains 200 lines, with three numbers (c_{kx}, c_{ky}, c_{kt}) on each line. The following datasets are provided:

Dataset	Components	Motion(s)	Mixing Proportion
mix.1	1	(0.5,1.5)	1.0
mix.2a	2	(2.0,1.0)	0.7
		(-1.0,0)	0.3
mix.2b	2	(2.5,1.0)	0.5
		(-1.0,-1.0)	0.5
mix.3	3	(-2.0, -0.5)	0.1
		(2.0,1.0)	0.7
		(0, 1.5)	0.2

Note: Each dataset has a corresponding Postscript file (mix.xx.ps) that shows the constraint lines plotted in the range $x = -4 \ldots 4$, $y = -4 \ldots 4$. You should see lines (roughly) intersecting at the motions shown in the table.

Optional: If you wish, you may plot the constraint lines in Matlab. If you choose to do this, you can display the solutions you find as marks on your plot.

(a) Least Squares Fitting

Write a Matlab program to minimize the error given by:

$$E(\vec{a}) = \sum_{k=1}^{K} \rho(\vec{c}_k \cdot (\vec{a}, 1)). \tag{3}$$

where $\rho(e) = e^2$. This finds the LSQ solution for the optic flow $\vec{a}$.

Show the results from the least squares approach when the data set consists of one and two distinct motions. Use the datasets `mix.1` and `mix.2a`. Comment on your results.

(b) Robust Fitting

Modify your program in part (a) to use the estimator

$$\rho(e;\sigma) = \frac{e^2}{\sigma^2 + e^2}. \tag{4}$$

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 (3). You should use the iterative method described in class based on successive application of a weighted least squares estimator. To demonstrate the algorithm show successive steps, either as marks on the plot of constraint lines, or by printing out the values of $\vec{a} = (v_x, v_y)$ at each step.

Use various values of σ between 0.2 and 2, and try your program on the for the datasets `mix.1`, `mix.2a`, and `mix.2b`. Also try various initial guesses. Comment on the quality of your results, and discuss the behaviour of your program for various values of σ.

For the multiple-motion cases, the robust estimator will usually find one of the motions (typically, whichever is closest to the starting point). To find a second motion, try the following: Once you find $\vec{a}_1$ for the first motion, remove all nearby lines (eg., $\|\vec{c}_k \cdot [\vec{a}, 1]^T)\| < 2\sigma_{meas})$) and fit the model to the remaining data.

(c) Mixture Models

Here we model multiple motions, $\vec{a}_n$, $n = 1 \ldots N$. The probability of a given constraint line is:

$$p(\vec{c}_k|\mathcal{A}, \Pi) = \sum_{n=1}^{N} \pi_n p(\vec{c}_k|\vec{a}_n) \tag{5}$$

where

$$\begin{aligned} p(\vec{c}_k|\vec{a}_n) &= Gaussian(\vec{c}_k \cdot \vec{a}_n|0, \sigma_v) \\ &= \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{1}{2\sigma_n^2}(\vec{c}_k \cdot (\vec{a}_n, 1))^2\right) \end{aligned} \tag{6}$$

and $\sum_n \pi_n = 1$.

Construct a mixture distribution containing two components. Fix the variance $\sigma_v = 0.25$. Adapt the velocities $\vec{a}_1$, $\vec{a}_2$ and the mixing proportions π_1, π_2 using the EM algorithm described in class. At each step of the EM algorithm, show the progress, either by plotting the velocities, or by printing them out. Show the final values for the velocities *and* mixing proportions and compare them to the table.

Demonstrate the mixture fitting using a two-component model on datasets `mix.1`, `mix.2a`, and `mix.2b`. Try several different starting conditions to demonstrate convergence. Comment on your results.

Finally, fit a mixture with *three* components to the dataset `mix.3`. **Note:** You will probably have probems reaching the global minima. If so, just show a few attempts, and the local minima achieved.

(d) Choosing Intial Conditions by Random Sampling

A good way to choose initial conditions for (b) and (c) is to randomly select constraint lines and see if they intersect at a common point. A simple algorithm is to choose three lines, $\vec{c}_1$, $\vec{c}_2$, and $\vec{c}_3$. Run the LSQ code from part (a) to find the intersection point $\vec{a}$. To determine if the lines intersect, check the average distance of the lines to the intersection point:

$$d_{avg} = \left(\frac{(\vec{c}_1 \cdot (\vec{a}, 1))^2 + (\vec{c}_2 \cdot (\vec{a}, 1))^2 + (\vec{c}_3 \cdot (\vec{a}, 1))^2}{3} \right)^{\frac{1}{2}} \tag{7}$$

A suitable threshold for d_{avg} depends on the measurement noise (σ_{meas}) and the range of velocities. For this exercise, use the threhsold $d_{avg} \leq \sigma_{meas} = 0.25$ to indicate when the lines intersect.

Repeat parts (b) and (c) using random sampling to choose initial conditions for the one and two component datasets.

For part (b), repeatedly select triplets of constraint lines until $d_{avg} \leq 0.25$. Use the intersection point $\vec{a}$ as the initial condition for the robust estimator. Demonstrate this method on `mix.2a`.

For part (c), we are looking for two or more different clusters. Find the first cluster as above. To find the second cluster, continue sampling triplets of points until you find a second cluster where $\|\vec{a}_2 - \vec{a}_1\| \geq 3\sigma_{meas} = 0.75$. In other words, assume that the clusters are well separated. Run the EM algorithm using these starting conditions. Try this method on `mix.2a` and `mix.2b`.

Bonus Problem: See if you can get the three component mixture to converge with good starting conditions.

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.