

CSE 152 Assignment 2  
Spring 2005  
Due Friday, April 29 by 5pm

In this assignment you will implement a program to track moving color blobs in a sequence of image frames.

**Step 1** Write a function `explore(image)` which implements the connected component algorithm described in class. You may assume 8-connectedness for your implementation. Your program should return a new image where each connected region is marked with a distinct positive integer. Test your program on the given input files `binary1.bmp` and `binary2.bmp`.

**Step 2** Now compute the moments and the central moments of each marked region. You should implement two functions, `moment(i, j, marked_image, number)` and `centralMoment(i, j, marked_image, number)`, which computes the  $i, j$  th moment and central moment given the marked image and the marker number.

**Step 3** On each image from Step 1, draw the centroid of each region as a circle. Then, compute the eigenvectors of the centralized second moment matrix and draw the two eigenvectors on the centroid. This should indicate the orientation of each region.

**Step 4** Next, implement the color selection procedure that converts the color image `erasers.bmp` into a binary image. First, convert your input image to HSV (you may use `rgb2hsv`). The input image has a blue eraser on a desk, so select target values for the hue and saturation that correspond to the blue object. Then, select the pixels exceeding a minimum intensity whose hue and saturation values lie within a neighborhood of the target.

Once you have your binary image, explore the connected components and compute the centroid and the eigenvectors of each marked region. Since there is noise in the binarization process, ignore color blobs that are smaller than a reasonable size, e.g. 100 pixels.

**Step 5** Finally, implement nearest-neighbor data association to track each blob as it moves from frame to frame. The metric we will use will be sum of squared differences (SSD). The idea here is that given a target blob  $T$  in frame  $i - 1$ , we want to find the blob  $B$  in frame  $i$  that most closely matches  $T$ . Each blob will have two criteria: size and location (x and y position). Let us denote the value of a criteria  $f$  for blob  $B$  at frame  $i$  to be  $f(B, i)$ . Then, the SSD metric is defined as

$$\sum_{f \in \{\text{criteria}\}} \alpha_f (f(B, i) - f(T, i - 1))^2$$

where the  $\alpha_f$  denote weights for each criteria. Thus, the blob that most closely matches  $T$  has the minimum SSD.

Test your algorithm on the movie `movingErasers.avi`. In this movie, since location is more important than size, set  $\alpha_{\text{size}} = 0.1$  and  $\alpha_{\text{location}} = 1.0$ .

## What to turn in

- For step 1, turn in your `explore.m` function. For each test image, turn in a set of binary images with each one containing a single segmented object (a single marked region) from the input image. Name them `binary1-1.jpg`, `binary1-2.jpg`, `binary 2-1.jpg`, `binary 2-2.jpg` and so on.
- For step 2, turn in your `moment.m` and `centralMoment.m` functions.
- For step 3, turn in the script `displayMoments.m` which displays the original images from step 1 with the centroids and eigenvectors. Make sure you plot each centroid with a different color. In addition, save your results as `binary1-result.jpg` and `binary2-result.jpg` and turn them in.
- For step 4, turn in your `colorSelect.m` script which converts `erasers.bmp` into a binary image. Plot the centroid and eigenvectors on top of the original image and save the result as `erasers-result.jpg` which you will turn in.
- Finally, for step 5, load the movie `movingErasers.avi` and process each frame. Instead of making a movie that shows the tracking results, simply plot the trajectory of the centroid and eigenvectors of each blob on a 2D plot. Turn in your plots as `movingEraser1.jpg`, `movingEraser2.jpg` and so on.
- Email your results to `wychang@cs.ucsd.edu` with the title CSE 152 Assignment 2. Please attach a single zip file.

## Tips

- Be careful how you implement the exploration and moment functions. Try to avoid using for loops as much as possible; you don't want to loop through each pixel in the image a billion times over. For the connected-component exploration, you may want to convert the algorithm to explore regions iteratively rather than recursively.
- To plot the centroids, use the `plot` command as follows:

```
plot(x, y, '.', 'markersize', 30, 'color', colors(k))
```

where `x`, `y` denote the location of the marker, and `colors` is an array that contains characters corresponding to different colors, e.g. `['r', 'g', 'b']` etc.

To plot the eigenvectors, use the `quiver` command:

```
quiver(x, y, vx, vy, 30, colors(k))
```

where `x`, `y` denote the starting location and `vx`, `vy` denote the x and y components of the vector.

Note that for both of these commands you may need to switch the x and y components to display it properly (due to how Matlab indexes the image matrices).

- To load a movie, use the `aviread(filename)` command, and to extract each frame of the movie, use `frame2im(mov(frame_number))`, where `mov` is a variable containing the movie.
- Get started as soon as possible!