In this assignment you will implement a program to identify moving objects in a sequence of image frames.

(a) Write a function `explore(image)` which implements the connected component algorithm described in class. You may assume either 4-connectedness for your implementation. Your program should take a binary image as input and 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`. [15 points]

(b) 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. [15 points]

(c) On each image from part (a), 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. Save your commands in a script named `displayMoments.m`. [5 points]

(d) Next, implement the segmentation algorithm that converts the image `snapshot.png` into a binary image. First, convert your input image to HSV color space (use `rgb2hsv.m`). You will use the hue channel to segment your image. Select a lower and upper threshold value that separates the background (the desk and the hand) from the three objects. Implement this as a function named `segment(image)`.
Once you have your binary image, explore the connected components and compute the position and orientation of each marked region. Since there is noise in the binarization process, ignore regions with area smaller than a reasonable size, e.g. 500 pixels. [5 points]

(e) Finally, classify the objects as “pencil,” “eraser,” “cap,” or “other” by thresholding the size and the orientation of each region. Once you have classified them, plot the position, orientation, and label of each region on the image `snapshot.png`. Using your classification algorithm, find and label these objects in each frame of the movie `pencil.avi` and plot the trajectory of each object. [10 points]

What to turn in

- For part (a), 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.png`, `binary1-2.png`, `binary2-1.png`, `binary2-2.png` and so on.
For part (b), turn in your `moment.m` and `centralMoment.m` functions.

For part (c), turn in your `displayMoments.m` which displays the original images from part (a) with the centroids and eigenvectors. Plot each centroid with a different color. In addition, turn in your results as images `binary1-result.png` and `binary2-result.png`.

For part (d) and (e), turn in your `segment.m` function which converts `snapshot.png` into a binary image. Plot the centroid, eigenvectors, and labels on top of the original image and turn in your result in `snapshot-result.png`.

Part (e): for each object, show your result by plotting (on a single plot) its position and orientation computed over the entire image sequence. Turn in your plots as images named `classify-eraser.png`, `classify-pencil.png`, etc., and save your commands in a script named `classify.m`.

Email your results to wychang@cs.ucsd.edu with the title CSE 152 Assignment 2. Please attach a single file in zip format. Note that there is no hardcopy portion in this assignment.

Tips

- Be careful how you implement the exploration and moment functions. Try to avoid using for loops as much as possible. To implement connected-component exploration efficiently, we can use an iterative algorithm rather than the recursive one discussed in class. A description can be found in Chapter 4 of “Robot Vision” by Horn.

- To plot the centroids, use the plot command as follows:

  ```matlab
  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:

  ```matlab
  quiver(x, y, vx, vy, 100, 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!