Visual Perception Course work

Abd El Rahman SHABAYEK
SnT, SIGCOM, Computer Vision Group, University of Luxembourg

March 19, 2017

1 Objective and Tools

There are three objectives of this course work:

1. to learn MATLAB as a powerful computer vision tool.
2. to learn one of the most important computer vision open source libraries, OpenCV.
3. to test and implement some important computer vision algorithms.

Don't panic !! Most of the requested tasks are implemented by few lines of code. You are expected to master these tools.

If you have any questions you should refer to:

1. Mathworks online documentation and available codes.
2. MATLAB documentation and demos.
3. OpenCV 2 Computer Vision Application Programming Cookbook.
4. OpenCV documentation.
5. Google it !
6. Send me an email: Abdelrahman.Shabayek@members.em-a.eu

Note that you will be evaluated on the level of your questions.

2 The projects

1. Create a MATLAB computer vision toolbox.

   Use "guideGUIDE" to build your GUI. Use MATLAB documentation, demos, and online help.

   For the calibration part in (4q), link your program to the following omni-directional calibration toolbox OCamCalib toolbox.

2. Create a Computer vision application using OpenCV.

   You will use OpenCV 3.2. You are free to use any IDE (Integrated Development Environment), however it is recommended to use Qt which is a free and easy to learn open source cross-platform IDE. You may also use Matlab with OpenCV, it is up to you !!

You should submit a **complete documentation** of each of your programs showing how to use it. You will present your work and show a demo of your program.

In each of the previous projects you have to interactively:

1. Input an Image, video, or a live stream from USB Camera and show it in a separate window named "Input".
2. Apply a requested process.
3. Show the output in a separate window named "Output". If the input is video or a live stream, the output should be in real time (both windows are side by side showing the input and output at the same time).
4. The GUI should provide the following options (you may add more if you wish!):
   - (a) Add salt and pepper noise.
   - (b) Show a logo at a corner of the input (ROI).
   - (c) Convert to a new colorspace.
   - (d) Compute the histogram.
   - (e) Equalize the histogram.
   - (f) Apply morphological operations (Dilate, Erode, Open, Close).
   - (g) Blur.
   - (h) Apply Sobel and Laplacian operators.
   - (i) Sharpen by applying a kernel.
   - (j) Edge detection using Canny.
   - (k) Extract lines and circles using Hough transform.
   - (l) Find contours of connected objects and draw them.
   - (m) Apply components' shape descriptors (bounding box, minimum enclosing circle).
   - (n) Extract corners using Harris and apply non-maximal suppression.
   - (o) Extract FAST, SURF, and SIFT.
   - (p) Find matches between two different views.
   - (q) Calibrate the attached camera, show the extracted corners of the calibration pattern in each image, and undistort the input image given the calibration results.

(r) Attach two cameras (or take two images from the same camera after moving the camera to a new position), compute the fundamental matrix using: (only one of the following in MATLAB project, and all of them in the OpenCV project)

7 point method, 8 point method, and RANSAC.

(s) Draw the epipolar lines and show the epipole in the two images.

(t) Compute the homography between two images (capture an image then apply a pure rotation to the camera, then capture another image).

(u) Add the two images connected by the found homography in one image (mosaic).

(v) Stitch two or more images (use the stitching module).

(w) Deep learn a set of faces and recognize them using this code.

(x) Save the result.

5. Group the previous options in different logical GUI entities (under menus or tabbed view or .....).

OpenCV Hint: Convert the input image into a gray-scale image before any operation. All the requested options are well explained in the reference book, although you may need to search in the documentation for more details.

If you have problems with processing a video or the camera input, make sure that all the options work on the image level then go back to the video/camera level. In concept everything should be the same with a few more lines of code to handle the video/camera input/output.

Be creative, design the program in your own way, try to add more options if you have time (you will get a bonus !!), assume it will be used by your company to apply different image processing and computer vision operations.

3 Marking

You will give a 20 min presentation on 29th *May* 2017 and submit your completed project. The total marks of the two projects is 100.

- MATLAB project: 40/100
- OpenCV project: 60/100