Project 1 of CSE 473/573

Due Time: 3PM Oct. 08 at Norton 112

Guidelines

1. Please write your programs using Python. Both Python 2 and Python 3 are OK (though we highly recommend using Python 3).
2. For task 1 and 2, you should not use OpenCV library (the only two exceptions are `cv2.imread()` and `cv2.imshow()`, two functions for reading and displaying an image.). You need to write the convolution code ON YOUR OWN (You should not use any other libraries, which provide APIs for convolution or correlation.). For task 3, you could use OpenCV library.
3. Images for all three tasks will be uploaded to Piazza on Monday (Sep. 17). You should ONLY use the images uploaded to Piazza to test you programs and obtain results that you are to include in your report. Fig. 1, Fig. 2 and Fig. 3 are for illustration only. You should not take screenshots of them and use the screenshots to test you programs and obtain any result that you are to include in your report.
4. Submit a project report of up to 15 pages (hard copy) by the due date. In the report, please provide the image results and the source code. You also need to explain the proposed method for task 3. Please upload the source code before the due date to Piazza for TA's check.
5. You need to work on this project independently and plagiarism will be penalized.

1 Edge Detection [5 points]

Figure 1: Image for edge detection

Write programs to detect edges in Fig. 1 (along both x and y directions) using Sobel operator. In your report, please include two resulting images, one showing edges along x direction and the other showing edges along y direction.

2 Keypoint Detection [5 points]

Write programs to detect keypoints in an image according to the following steps, which are also the first three steps of Scale-Invariant Feature Transform (SIFT).
1. Generate four octaves. Each octave is composed of five images blurred using Gaussian kernels. For each octave, the bandwidth parameters σ (five different scales) of the Gaussian kernels are shown in Tab. 1.
2. Compute Difference of Gaussian (DoG) for all four octaves.
3. Detect keypoints which are located at the maxima or minima of the DoG images. You only need to provide pixel-level locations of the keypoints; you do not need to provide sub-pixel-level locations.
In your report, please (1) include images of the second and third octave and specify their resolution (width $\times$ height, unit pixel); (2) include DoG images obtained using the second and third octave; (3) clearly show all the

Figure 2: Image for keypoint detection

Octave					
1st	$\frac{1}{\sqrt{2}}$	1	$\sqrt{2}$	2	$2\sqrt{2}$
2nd	$\sqrt{2}$	2	$2\sqrt{2}$	4	$4\sqrt{2}$
3rd	$2\sqrt{2}$	4	$4\sqrt{2}$	8	$8\sqrt{2}$
4th	$4\sqrt{2}$	8	$8\sqrt{2}$	16	$16\sqrt{2}$

Table 1: The bandwidth parameters σ (five different scales) of the Gaussian kernels used in the first step of keypoint detection.

detected keypoints using white dots on the original image (4) provide coordinates of the five left-most detected keypoints (the origin is set to be the top-left corner).

3 Cursor Detection [5 points + 3 bonus points]

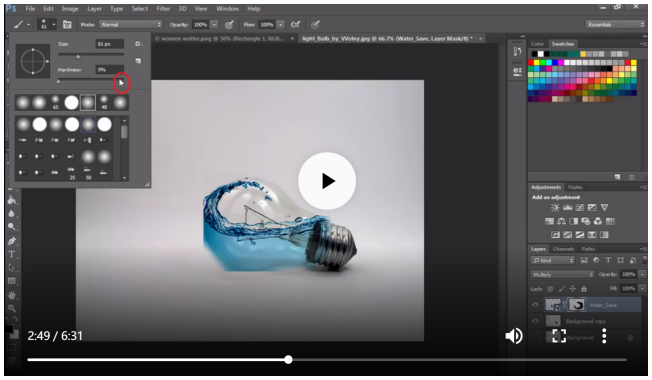

Figure 3: Illustration of cursor detection task, which aims to locate the cursor highlighted in the red circle.

For the task of cursor detection, which aims to locate the cursor in an image, two sets of images and cursor templates, named as "Set A" and "Set B", will be provided to you. Set A is composed of a total number of 25 images and 1 cursor template. Set A is for task 1., i.e., the basic cursor detection which contributes to 5 points. Set B is composed of a total number of 30 images and 3 different cursor template. Set B is for task 2., i.e., which contributes to 3 bonus points.
1. Detect cursors in Set A. [5 points]
2. Detect cursors in Set B, which is more challenging. [3 bonus points]

Note that we will randomly select and run the code you submitted on a set of 20 withheld images to test the performance of your cursor detection programs.