Intro to Image Understanding (CSC420)

Assignment 1

Posted Sept Oct. 12, 2016; Submission Deadline: Oct. 21st 11.59pm 2016

Instructions for submission: Please write a document and submit a **PDF** with your solutions (include pictures where needed). Include your code inside the document, and submit through **MarkUS**.

For full marks you must show your work, not just your final answer.

Max points: 13, max extra credit points: 3

1. Interest point detection:

   (a) **[2 points]** Write your own function for Harris corner metric using the harmonic mean (slide 29, lecture 6). Display your result for the attached image **building.jpg** showing your cornerness metric output. You can use built-in functions for convolution, gradients, but you must compute M yourself. Adjust α to get a good result.

   (b) **[2 points]** Write your own function to perform non-maximal suppression using **ordfilt2.m** or your own morphological operators function of choice. Use a circular element, and experiment with varying radii r as a parameter. Explain why/how the results change with r.

   (c) **[2 points]** Write code to search the image for scale-invariant interest point (i.e. blob) detection using the Laplacian of Gaussian and checking a pixel's local neighbourhood as in SIFT. You may use code from **tutorial 4** as a starting point. You must find extrema in both location and scale. Find the appropriate parameter settings, and display your keypoints for **synthetic.png**. *Hint: Only investigate pixels with the LoG above or below a threshold.*

   (d) **[1 point]** Compare and contrast the Harris corner metric with non-maximal supression as a keypoint detector to the Laplacian of Gaussian method. Show examples where they detect different keypoints and the same keypoints and explain why they are the same/different using **synthetic.png** and **building.png**.

2. For this question you will use interest point detection for matching using SIFT. You may use a SIFT implementation (e.g. http://www.vlfeat.org/), or another, but specify what you use.

   (a) **[0.5 points]** Extract SIFT keypoints and features for **book.jpg** and **findBook.jpg**.

(b) **[1.5 points]** Write your own matching algorithm to establish feature correspondence between the two images using the reliability ratio on Lecture 8, slide 23. You can use **pdist2.m**, but you must find the matches yourself. Experiment for different thresholds.

(c) **[2 points]** Affine transformation: Use the top k correspondences from part (b) to solve for the affine transformation between the features in the two images via least squares using the Moore-Penrose psudeo inverse. Demonstrate your results for various k. Use only basic linear algebra libraries.

(d) **[0.5 point]** Visualize the affine transformation. Do this visualization by taking the four corners of the reference image, transforming them via the computed affine transformation to the points in the second image, and plotting those transformed points. Please also plot the edges between the points to indicate the parallelogram. If you are unsure what the instruction is, please look at Figure 12 of [Lowe, 2004].

(e) **[1.5 points]** Write code to perform matching that takes the colour in the images into account during SIFT feature calculation and matching. Explain the rational behind your approach. Use **colourTemplate.png** and **colourSearch.png**, display your matches with (2.d).

3. **Extra: [3 points] total** (this is an optional exercise) Implement your own SURF feature descriptor, which is very similar in steps to SIFT. You can use any keypoint localization algorithm of your choice. See: http://www.vision.ee.ethz.ch/ surf/eccv06.pdf and or Algorithm 6 in http://www.ipol.im/pub/art/2015/69/.

   You may submit up to two days late without penalty if you do well in this exercise.

   (a) **[2 points]** Your descriptor must include:

   - Build your own Harr-wavlet filters
   - Computation of Harr-wavlet responses
   - Obtain the orientation using a circular neighbourhood of radius 6s.
   - Gaussian weighting
   - Rotation and scale invariance via feature window re-orientation and scale fitting (you can use bilinear interpolation instead of integral images)
   - Feature sub-binning into a 4x4 block of regions and 16 features per region

   (b) **[1 points]** Compare your algorithm to SIFT for the **book.jpg** and **findBook.jpg** pair, and at least two additional of your own image pairs, and discuss the pros/cons in accuracy. Don't worry about the speed of your implementation.