ECE 5460

Image Processing

Fall 2023

Project 2

Posted: October Monday 9, 2023 **Due:** Wednesday 23:59, October 25, 2023

We recommend you use the basic Python notebook discussed in the tutorials. You can run it locally or on Google Colab. To use Colab, upload it to Google Drive and double-click the notebook (or right-click and select Open with Google Colaboratory), which will allow you to complete the problems without setting up your own environment. Once you have finished, make sure all the cells are run before downloading the notebook).

Submission Instructions: Please submit three categories of files on Canvas: (1) the Python notebook with the relevant source code and with all the cells run, and (2) The input output images in jpeg format (3) A typed !! (latex, word, etc) report saved as as a pdf file that includes all input and output images and a brief description the processing details and description of each input/outputimage

Late Submission Policy: As discussed in the late policy. Due dates will be strictly enforced. 20% penalty for every late day. Solutions will be posted to the course web page in 5 days. After solutions are posted, no credit will be issued for late work.

Problem 1 *Scale Space Blob Detection*

Algorithm outline

1. Generate a Laplacian of Gaussian filter.
2. Build a Laplacian scale space, starting with some initial scale and going for n iterations:
   - Filter image with scale-normalized Laplacian at current scale.
   - Save square of Laplacian response for current level of scale space.
   - Increase scale by a factor k.
3. Perform nonmaximum suppression in scale space.
4. Display resulting circles at their characteristic scales.

For creating the Laplacian filter, use the `scipy.ndimage.filters.gaussian_laplace` function. Pay careful attention to setting the right filter mask size. You have to choose the initial scale, the factor k by which the scale is multiplied each time, and the number of levels in the scale space. I typically set the initial scale to 2, and use 10 to 15 levels in the scale pyramid. The multiplication factor should depend on the largest scale at which you want regions to be detected.

To perform nonmaximum suppression in scale space, you should first do nonmaximum suppression in each 2D slice separately. For this, you may find functions `scipy.ndimage.filters.rank_filter` or `scipy.ndimage.filters.generic_filter` useful. Play around with these functions, and try to find the one that works the fastest. To extract the final nonzero values (corresponding to detected regions), you may want to use the `numpy.clip` function. You also have to set a threshold on the squared Laplacian response above which to report region detections. You should play around with different values and choose one you like best. To extract values above the threshold, you could use the `numpy.where` function.

For choosing the appropriate sigma values, note that the radius and standard deviation of a Gaussian such that the Laplacian of Gaussian has maximum response are related by the following equation: $r = \sigma\sqrt{2}$.

Problem 2 *Difference of Gaussian pyramid* Insteadofcalculating a LoG, we can often approximate it with a simple Difference of Gaussians (DoG). Specifically many systems in practice compute their ?Laplacian of Gaussians? is by computing $(I * G_{k\sigma}) - (I * G_\sigma)$ where G_γ denotes a Gaussian filter with a standard deviation of γ and $k > 1$. If we want to compute the LoG for many scales, this can be far faster, rather than apply a large filter to get the LoG, one can get it for free by repeatedly blurring the image with little kernels. Implement the difference-of-Gaussian pyramid that we discussed in the SIFT lectures and described in David Lowe's paper [1] posted on the Carmen. Compare the results and the running time to the direct Laplacian implementation.

For extra credit, in your report discuss why computing $(I * G_{k\sigma}) - (I * G_\sigma)$ might successfully roughly approximate convolution by the Laplacian of Gaussian. You should include a plot or two showing LoG and DoG filters in your report.

Acknowledgment: Project based on versions developed by James Hays, David Fouhey, Noah Snavely, Svetlana Lazebnik and Derek Hoiem.

References

[1] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.