Project Proposal ECE1782

**SUPERPIXELS**

Group: Eugene Osovetsky (eugene.osovetsky@gmail.com), Felipe Sander Pereira Clark (felipe.clark@gmail.com), Harika Gaggara (harika.gaggara@mail.utoronto.ca)

Superpixels [1] is a technique that groups pixels in an image together based on a proximity criterion that is perceptually meaningful. The end result is a simplified “mosaic” representation of an image that still preserves the visual semantics of the image. Fig. 1 shows an example of an image segmented into 400 superpixels. The technique is normally used as a preprocessing step in computer-vision pipelines, but can be also used for artistic purposes. In the former case, it is typically used to segment and label pieces of an image or to track a subject of interest in a sequence of images.

Whichever the use of this technique might be, it is important that it does not slow down the pipeline into which it is inserted. For this reason, we believe that it is worth implementing this technique in GPUs. As an extra motivation to pick this project, the resulting algorithm might be valuable for Felipe in his workplace.



*Figure 1: Image Segmented into 400 superpixels.*

The actual algorithm we will implement is called “SLIC SuperPixels” and is described in [2]. It is essentially a “K-means” algorithm. In simplified form, the algorithm is:

1. Convert RGB image to LAB color space
2. Initialize superpixel centers (start out with them evenly distributed)
3. **Repeat until convergence (or for a bounded number of iterations):**
   1. **For each pixel, find the superpixel center “closest” to it, using a Euclidian-like distance metric in 5D space (L/A/B color and X/Y coordinates). The pixel is then marked as “owned” by that superpixel centre.**
   2. **For each set of pixels “owned” by one superpixel centre, compute the mean of all 5 coordinates (L/A/B color and X/Y), and then move the superpixel center to that mean.**
4. Post-process the image for display (enforce connectivity of pixels belonging to various superpixel centres, and color each pixel “owned” by a given superpixel centre with the superpixel centre’s mean color).

To keep the project reasonably-sized, we will focus on optimizing for performance only the two critical (repeated) kernels, **c(i) and c(ii) above**. We will do the minimum amount of work possible on the other steps, including the option of either writing sub-optimal kernels, implementing them on CPU, or omitting all or part of step (d) entirely (though we would really like to get to a full, working implementation producing pretty pictures!)

This project, thus scoped, is reasonably sized and the complexity of the algorithm is reasonable enough to be dealt with in the time frame we have until the end of the course. Also, we have found papers that present efficient CPU implementations [3] and others that claim significant performance gains in GPUs [4], so we have a reference to compare our own performance improvements gains.

**References**

|  |  |
| --- | --- |
| [1] | J. S. a. J. Malik, "Normalized Cuts and Image Segmentation," *IEEE,* vol. 22, no. 8, pp. 888-905, 2000. |
| [2] | A. S. K. S. A. L. P. F. a. S. S. : Radhakrishna Achanta, "SLIC Superpixels," EPFL, 2010. |
| [3] | A. S. K. S. A. L. F. a. S. S. Radhakrishna Achanta, "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods," *IEEE transactions on pattern analysis and machine intelligence,* vol. 34, 2012. |
| [4] | V. A. P. I. D. R. Carl Yuheng Ren, "gSLICr: SLIC superpixels at over 250Hz," *ArXiv ,* 2015. |