LENFEST SUMMER GRANT 2011 REPORT

Date submitted: **October 10, 2011**

Name: **Joshua Stough, Assistant Professor of Computer Science**
Project Title: **Visual Object Recognition in Natural Images**
Project Dates: **June 20, 2011 through August 26, 2011 (10 weeks)**
Project Location(s): **Washington and Lee University**

In Computer Vision research, object recognition refers to the task of automatically determining the presence of a specific object or class of object in an image. During this Lenfest research period, I continued my work in object recognition from the previous summer with three RE Lee scholars. We worked on two aspects of our automatic object recognition software, speed and accuracy. First, motivated by the limiting computational power I observed last summer, I worked with my students on obtaining speedup in our software. We achieved a greater than 20-fold speedup through the use of lower-level (faster) programming languages (with Paul Nguyen '13), and by exploiting the inherent parallelism in the problem through distributed computing (with Lee Davis '13). In this vein, one lasting contribution of this summer's work should prove to be Garrett Koller's '14 deployment of the Condor distributed computing platform (under my supervision), through which the Computer Science department's lab machines can perform general-purpose computation during their idle time. We also worked to improve our methodology to obtain more accurate results, implementing and testing various distance metrics. In the following Lenfest summary, I will briefly review the project and our proposed and achieved enhancements. I will conclude with results and future directions of focus.

In human perception there are thought to be several levels of abstraction to object recognition: the localization of interesting features (corners and the locations of contrast in general), the organization and simple geometry of these features, and at the highest level of abstraction the classification of these features with respect to our memory. Given the huge amount of image data generated today, automated object recognition is a common goal in computer vision research [1].

The PASCAL Visual Object Classes Challenge is a competition held each year in conjunction with an international computer vision conference [2]. The goal is to automatically recognize objects from about twenty classes in natural scenes (photos taken from the online flickr website). The classes include person, dog, airplane, bus, chair, bottle, and other animals, vehicles, and objects seen indoors. In the competition, all submitting groups' software systems are provided the same already-classified images for use as training data (that is, each training image is labeled according to the presence of the object classes). Generally, each object class is trained separately. Subsequently, the systems are rated according to their efficacy in classifying a large collection of unlabeled images. Many publications cite results on a particular iteration of the VOC Challenge database, which now consists of some 30,000 images.

Many of the successful methodologies for working with the large VOC dataset have a high computational cost, some requiring 5000 hours of training, or seven months on a single machine. There is a high degree of parallelism in the algorithms however; various statistics are computed from or applied to thousands of images independently. Given the computational cost, my major Lenfest goal was to better exploit the inherent parallelism in the problem. I did this with both the DistributedPython [4] package I authored and used with Lee Davis and the Condor deployment that I worked on with Garrett Koller [5].

My DistributedPython package makes large-scale parallel computing easily applicable to any task consisting of many independent parts—such as, for each image or for each object-type. Much distributed computing software is in the end, under many layers, a simple concept of asking another computer to perform your job (your program that needs to run). In distributed computing platforms, a pool of other computers is maintained along with a queue of jobs to be done, so that many jobs can be run independently in parallel. DistributedPython accomplishes this in as few layers as practicable. Using the relatively easy Python programming language, at the top level one simply specifies a list of commands and requests they be executed in parallel, and everything else is taken care of. Lee Davis used this code to achieve a speedup equal to the number of machines harnessed.

There is some danger in DistributedPython in that if I send a job to one of the lab machines, then someone at that machine may notice a slow down as I hog the resources. Condor is a popular distributed computing platform with more versatile job control mechanisms to account for distributed ownership of the resources. For example, Condor has to ability to transfer jobs from one machine that becomes busy to another that is still idle. Garrett Koller '14 led our deployment of Condor this summer, customizing and administering it to use our new advanced-lab machines. As Garrett continues as a work-study this fall, we are advertising Condor's availability and our guidance to other faculty and students with computational needs. Condor figures heavily in a workshop proposed by me for Prof. I'Anson's IQ center grant. I am also attending the upcoming supercomputing conference (called SC11) with Garrett, where he will present a poster describing his achievements.

In the Lenfest grant, I also received $1500 for a networked-attached storage (NAS) device. This device serves as a common file system (or file server) for the compute nodes in the Condor cluster. This is where the tens of thousands of images are stored for VOC Challenge, for example. By having the NAS, it is much easier to organize parallel jobs. Also, the NAS serves as the common location for all configuration files related to Condor, making modifications and updates to the platform much easier.

While DistributedPython and Condor are convenient ways to parallelize across images and objects, there are some tasks in our object recognition pipeline where speedup is only achievable at a lower level. Paul Nguyen obtained speedups greater than 50 on these tasks, using the C programming language to complement Matlab and exploiting low-level parallelism through the use of pthreads and openmp, which distribute jobs to the multiple cores in a multicore system.

I am grateful for the opportunity that the Lenfest grant has afforded me this summer. While the results are not improved by enough to reach the top-tier of the VOC challenge, I am able to get feedback on my ideas much faster. Further, these experiences in high-end scientific computing will be extremely valuable to these students. I am currently continuing this research, and I look forward to improving our position in future competitions and submitting this work for publication.

References:

1. D. Lu and Q. Weng. A survey of image classification methods and techniques for improving classification performance. *International Journal of Remote Sensing*, 28(5):823–870, 2007.
2. Everingham, M. et al. The PASCAL Visual Object Classes Challenge 2010 Results. http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2010/results/index.html#KEY_WLU_SPM_EMDIST
3. Yang, J., Yu, K., Gong, Y., Huang, T. “Linear Spatial Pyramid Matching Using Sparse Coding for Image Classification ”, Int. Conf. On Computer Vision and Pattern Recognition (CVPR), 2009.
4. Publicly available codebase: http://code.google.com/p/distributed-python-for-scripting/
5. Condor@W&L: http://condor.cs.wlu.edu