Encouraging the Development of Undergraduate Researchers in Computer Vision

Clark F. Olson
University of Washington, Bothell
Computing and Software Systems, Campus Box 358534
18115 Campus Way NE, Bothell, WA 98011-8246 USA

ABSTRACT

In a small computer science department without a graduate program, it is sometimes difficult to attract research students. This is particularly true for research in computer vision, since it is built upon a substantial body of knowledge, including considerable mathematics, that most undergraduates are not familiar with. My approach to encouraging students to take part in this research starts by introducing computation with images in early programming classes. Students become comfortable working with images in a structured framework, where they are not exposed to excessive underlying details. The students that become interested in working with images can take my computer vision class. This course is taught in a way that students can understand the material without having a deep background in mathematics. Students that are successful in this class are ready for (and encouraged to) work on undergraduate research projects and perform internships in computer vision research. While my strategy focuses on computer vision, similar approaches could be used for other research areas.

Categories and Subject Descriptors

K.3 [**Computers and Education**]: General

General Terms

Algorithms, Human Factors

Keywords

Computer Vision, Pedagogy, Research

1. INTRODUCTION

Collaborating with undergraduate students in faculty research has the potential to be beneficial to both students and faculty. Usually, the benefit is greater for the student, since undergraduates are at a stage of development where they require considerable coaching to perform research, but can derive life-long learning skills from the experience.

ITiCSE'06, June 26–28, 2006, Bologna, Italy.

Encouraging students to get involved and preparing them so that they don't require constant supervision is a challenge. Unfortunately, many students view the university as a place where knowledge is fed to them by their professors, rather than a place of discovery. For a researcher in computer vision, there is an additional challenge, since there is a wide breadth of knowledge necessary to be fully conversant in the field. Techniques common in computer vision are drawn from several areas, including optics, set theory, signal theory, probability, projective geometry and differential geometry, among others. It is common for computer vision courses to have prerequisites such as data structures, image processing, calculus, linear algebra, and numerical methods [1].

I have adopted a strategy where students are introduced to images in an early programming class. Our department currently serves only upper-division students, with students performing their lower-division work at community colleges or four-year universities. This means that the earliest opportunity for me to introduce students to images is in our first programming course at the junior level, although there is no reason that it could not be introduced earlier in other universities (and, indeed, is [2, 5, 6, 8]). Other examples of using images in core computer science courses can be found in [3, 10, 12]. Not all of our students get this introduction, since I am one of several faculty that teach this course. However, many of the students have gone on to take the computer vision class without this introduction.

Students who are particularly interested in this subject take my elective course in computer vision. This course is structured in a way to make many aspects of computer vision accessible to students who don't have an extensive mathematical background. I focus the course on a set of programming assignments that give the students experience with implementing computer vision techniques.

Having completed this course, students have the opportunity to work on independent study, undergraduate research, and internships using their computer vision knowledge. A ten-credit internship is required to complete a degree in our department. Most students choose to fulfill this requirement by working in industry, but they have the option to work with a faculty member on research instead.

2. PROGRAMMING WITH IMAGES

The first step in my strategy to prepare and motivate students for future work in computer vision is to introduce them to programming with images in our first (junior-level) programming course. One goal of this course is to integrate mathematical principles with programming topics. Discrete mathematics are given equal time with basic algorithms and abstract data types. Since the students arrive with very different backgrounds, another goal is to put all of the students on equal footing for the rest of the curriculum. The only assumption that I can make about the background of these students is that they have taken the prerequisites to our department, which include two quarters of programming, two quarters of calculus, and one of statistics. To get these students started, I give them a C++ library that includes an image data structure (not object-oriented) and functions for reading an image from a file, saving an image to a file, as well as creating, copying, and deallocating an image.

The first assignment, given on the first day of class, is to write a simple program that reads an image, creates a mirror-image of it, and changes the colors by pseudo-random amounts. There are two goals to this assignment. The first is to get students comfortable programming with images. The second is for them to notice that, when they save the image, the file stores colors that are not quite the same as the ones that were stored in memory. The reason for this is lossy compression that takes place in the file format (I use GIF files, but almost any format could be used).

One major theme of the class is the use of abstraction, for both data and functionality. The students gain experience programming with abstraction by creating an image class the wraps around the provided image data structure and functions. This makes images first-class objects and provides a stable interface that abstracts away from the underlying data structure. After creating the new class, students use the class to perform a task similar to the first program, such as creating an image photo-negative (see Fig. 1). This allows them to see the differences in use between the programming styles. While the lessons learned here primarily have to do with abstraction and object-oriented programming, the students gain experience with images.

The remaining assignments focus on particular topics in the class, such as recursion and linked-lists, but use images in a way that interests and motivates the students, while performing progressively more complex operations on images.

The third assignment helps the students in understanding recursion. Students create an image fractal by recursively dividing an image into quadrants. Two quadrants are filled with a smaller version of the input image and two quadrants are filled with a recursively defined image (see Fig. 2). This is an interesting programming assignment, since it is difficult to program without using recursion, but does not require a large program stack.

The final program in this course is to perform a simple image segmentation using both recursion and linked-lists by performing a depth-first search on the image pixels. Image segmentation is an important topic in computer vision, but a superficial algorithm is used in this class that focuses on recursion and linked-lists. Adjacent pixels with a similar color are accumulated in a linked-list and further neighbors are processed recursively until no similar neighbors remain. When this occurs, a new starting pixel is found and the process is repeated until all of the image pixels are in one of a set of non-overlapping groups. The image is then recolored, with each pixel being replaced by the average color in the group that it belongs to (see Fig. 3). Each group corresponds to a segment of the image sharing a similar color. While the techniques used do not yield a high-quality segmentation, students can observe the richness (and complexity) of the field.

3. COMPUTER VISION COURSE

I teach computer vision as an advanced undergraduate elective in our department. The only prerequisite for this course is data structures and algorithms (which succeeds the introductory programming course described above). No image-based assignments have been used to date in the data structures and algorithms course.

In my computer vision class, the largest obstacle is the mathematical background necessary to understand important computer vision techniques. Most students arrive having taken only two quarters of calculus and one quarter of probability and statistics. Some of the students will have seen linear algebra and coordinate transformations in our graphics course, but many will not. Some will have no familiarity with even matrix multiplication and most will not have seen partial derivatives, Fourier transforms, or singular value decomposition in their coursework. I give a two hour overview of linear algebra and spend some time discussing partial derivatives, but leave more complex topics, such as eigenvalue analysis, uncovered.

Given this lack of mathematical sophistication, one important issue is selecting an appropriate textbook. Most computer vision textbooks assume that the reader has a strong background in math. I have struggled finding a balance between coverage of the material and difficulty of the textbook. The first time I taught the course, I selected the recent textbook by Forsyth and Ponce [4], since it has strong coverage of the topics I present in the class. However, the students found this book too difficult, requiring much sophistication in math, and too theoretical. For the second offering of this course, I changed to a less recent book by Trucco and Verri [13]. While this book had less coverage, it covered most of the topics I discuss, and I felt it would be easier for the students to understand. Unfortunately, the level of math assumed was still at a high level. In the most recent offering of the course, I used the textbook of Shapiro and Stockman [11]. This textbook is better oriented towards computer science students, with good coverage, and an acceptable level of math. It definitely had the best response from the students.

Another issue to be considered is the approach to presenting the material. Maxwell [7] has pointed out that most computer science students have more experience with algorithms than with mathematics. He advocates presenting material from an algorithmic viewpoint, describing techniques using a process, rather than an equation. I follow this advice. The presentation of convolution, a technique for linear filtering, is a good example. Convolution is often introduced using an equation. For example, in the Forsyth and Ponce textbook [4] (and many others) it is introduced mathematically:

$$R_{ij} = \sum_{u,v} H_{i-u,j-v} F_{u,v} \tag{1}$$

This is not straightforward to understand intuitively. Computer science students tend to understand convolution better when it is described as an algorithm. It can be explained that the convolution kernel H is compared to the image F pixel-by-pixel for some relative location of the kernel, with

Figure 1: Creating an image negative. This is given as a straight-forward program to test the students abstraction of the image data structure. The input image in on the left and the desired output is on the right.

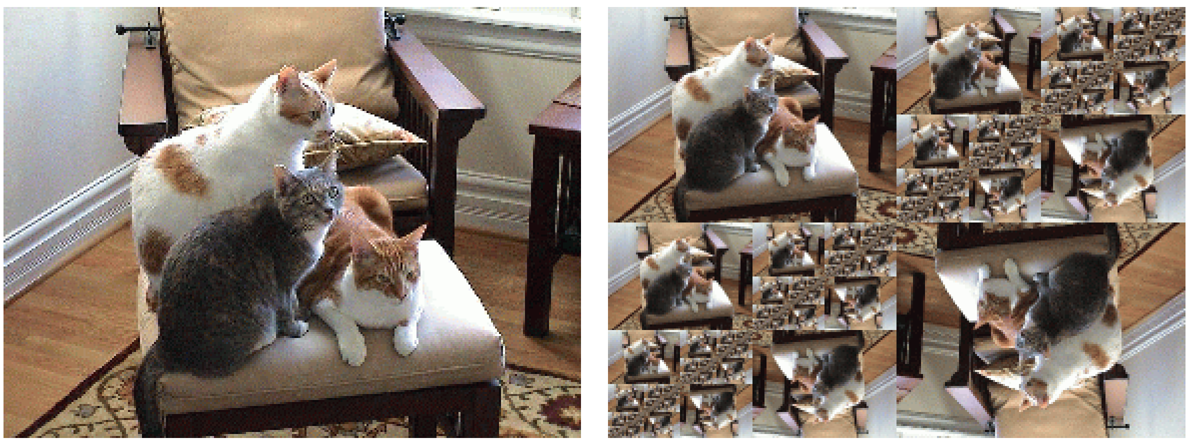

Figure 2: Image fractal assignment. This assignment forces students to use recursion in an interesting way. The input image in on the left and the desired output is on the right.

Figure 3: Image segmentation assignment. The goal of this assignment is to combine recursion and linked-lists. The input image in on the left and the desired output is on the right.

respect to the image, to calculate one value in the output R. Since u and v are negated in indexing H, this results in the kernel being flipped vertically and horizontally prior to the comparison.

The field of computer vision is quite broad and not all of it can be covered in a year, let alone a quarter. Instructors must decide which topics are most important to them and their students. Since my own work is largely in object recognition and mobile robot navigation, this influences my choice of topics. In the first half of the course, I concentrate on 2D techniques, including linear filtering, edge and line detection, and segmentation. This culminates with a discussion of the application of these ideas to retrieval from image databases. The second half of the course examines 3D techniques, such as stereo vision and motion estimation, as well as 3D object recognition. This is concluded by a discussion of vision techniques for mobile robot navigation. In this way, I can cover many important topics in computer vision, including those most important to my research, and bring them together in two current applications of computer vision (image databases and mobile robot navigation).

I have several goals in this course, but the most important are giving students a feel for the richness of the field and developing their image programming skills by solving problems and implementing algorithms. To achieve this, a large portion of their grade is based on five programming projects throughout the course. The first, relatively easy project, involves image warping using linear transformations. The remaining programs become progressivly more interesting and deal with edge detection, segmentation, stereo vision, and object recognition. I also assign two written assignments that amplify some of the fundamental concepts in the course.

Figures 4 and 5 illustrate two of the programs that the class implements. The first determines the presences of edges in an image, characterized by changes in the image color or intensity. The second performs image segmentation using k-means clustering. Fig. 5 shows an example of the input and output for the image segmentation problem. The goal of this program is similar to the last programming project in the programming class, but more sophisticated techniques are used to achieve a better segmentation. In this program, the students must use the color, location, and visual texture of each image pixel to determine the pixel groupings.

4. ADVANCED WORK

Most of the students in our department choose to complete their required internship with a local company. However, they have the option of working with a faculty member on a research project to complete the internship. Some students are also interested in undergraduate research projects prior to their internship. It has been my goal to interest some of our students in working with me on research projects in computer vision. I have been somewhat successful.

Over the past four years, seven undergraduate students have worked with me on either individual research for credit or internships in computer vision research. The projects these students worked on were all interesting, and each of them produced useful results. However, not all of them ended with a completely satisfactory resolution, since the students sometimes graduated or moved onto other work.

The most successful project was performed by a student who integrated research code into a general software architecture for mobile robots [9]. This project was highly useful to both the student and me. The student gained an introduction to research and experience developing code in the context of a large project and I was rewarded with a well-designed and integrated code set that is still in use for robot vision research. This student went on to win our President's Medal, the highest award for academic achievement at our campus. The other projects involved aspects of my ongoing research, with the following goals:

- Develop a technique for finding crystal stuctures in polarized images.
- Investigate the use of random sampling techniques in determining the motion of a camera by examining images taken by the camera at two positions.
- Determine the optimal parameters settings for performing robust matching between multi-modality images.
- Study the failure modes of a vision algorithm and improve the efficiency with respect to time and memory usage.
- Develop an image motion estimation and segmentation method to determine discrete object with different motions in an image sequence.
- Implement and investigate the efficacy of a robust feature extraction and matching method.

While this sample size of students is too small to generalize from, several students have indicated that using the image-based assignments in the early programming class motivated them to gain a further understanding of the topic.

5. SUMMARY

Getting undergraduates involved in research can be mutually beneficial. However, this can also take up significant faculty time and become frustrating for the student if they are not sufficiently prepared for their work in the project. Our department faces an additional handicap in that we serve only upper-division undergraduate students. This limits the amount of time we have to prepare students and achieve useful results in collaboration with them. I have taken an approach to preparing students for work in computer vision that starts early by using image-based assignments in our first programming class. After another quarter of programming, students have the opportunity to take my class on computer vision, where they learn to solve interesting problems through programming with images in C++. The final goal is for them to be able to perform advanced projects under my supervision. While some students have been able to accomplish this, I am still looking for ways to facilitate this process.

6. REFERENCES

[1] G. Bebis, D. Egbert, and M. Shah. Review of computer vision education. *IEEE Transactions on Education*, 46(1):2–21, February 2003.

[2] H. J. Fell and V. K. Proulx. Exploring Martian planetary images: C++ exercises for CS1. In *Proceedings of the SIGCSE Technical Symposium on Computer Science Education*, pages 30–34, 1997.

(a) (b)

Figure 4: Edge detection is performed with linear filtering, followed by thresholding, and non-maxima suppression. (a) Input image. (b) Output image.

(a) (b)

Figure 5: Image segmentation using k-means clustering. (a) Input image. (b) Output image.

[3] E. Finak and M. Heath. Image-processing projects for an algorithms course. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(5):859–868, August 2001.

[4] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall, 2003.

[5] G. Holmes and T. C. Smith. Adding some spice to CS1 curricula. In *Proceedings of the SIGCSE Technical Symposium on Computer Science Education*, pages 204–208, 1997.

[6] R. Jiménez-Peris, S. Khuri, and M. Patino-Martinez. Adding breadth to CS1 and CS2 courses through visual and interactive programming projects. In *Proceedings of the SIGCSE Technical Symposium on Computer Science Education*, pages 252–256, 1999.

[7] B. A. Maxwell. Teaching computer vision to computer scientists: Issues and a comparative textbook review. *International Journal of Pattern Recognition and Artificial Intelligence*, 12(8):1035–1051, August 1998.

[8] R. E. Montgomery. Image analysis: A group assignment in programming with breadth. In *Proceedings of the ASEE/IEEE Frontiers in Education Conference*, volume 2, pages 4d3.12–4d3.14, 1995.

[9] C. F. Olson, H. Abi-Rached, M. Ye, and J. P. Hendrich. Wide-baseline stereo vision for Mars rovers. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1302–1307, October 2003.

[10] S. Sarkar and D. Goldgof. Integrating image computation in undergraduate level data-structure education. *International Journal of Pattern Recognition and Artificial Intelligence*, 12(8):1071–1080, 1998.

[11] L. G. Shapiro and G. C. Stockman. *Computer Vision*. Prentice Hall, 2001.

[12] D. E. Stevenson. Image related applications for a core algorithms course. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(5):845–857, August 2001.

[13] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.