Teaching Image Computation: From Computer Graphics to Computer Vision

Bruce A. Draper and J. Ross Beveridge

Department of Computer Science

Colorado State University

Fort Collins, CO 80523

draper@cs.colostate.edu

ross@cs.colostate.edu

Keywords: Computer Vision, Computer Graphics, Education, Course Design
Abbreviated Title: From Graphics to Vision

Abstract

This paper describes a course in image computation that is designed to follow and build off an established course in computer graphics. The course is centered on images: how they are generated, manipulated, matched, and symbolically described. It builds on student's knowledge of coordinate systems and the perspective projection pipeline. It covers image generation techniques not covered by the computer graphics course, most notably ray tracing. It introduces students to basic image processing concepts such as Fourier analysis. It then introduces them to basic computer vision topics such as principal components analysis, edge detection, and symbolic feature matching. The goal is to prepare students for advanced work in either computer vision or computer graphics.

1. Introduction

Digital imagery, once considered an obscure aspect of computer science and electrical engineering, has become a pervasive part of modern computing. Beyond the obvious applications in entertainment and scientific visualization, digital images have become a central component of net-centered computing, human/computer interfaces, and databases, as well as data analysis for domains such as biometrics and environmental studies. Images are also relevant to the study of networking, encryption/compression, and parallel computing, which are more and more often used to transport, encode/compress, and process images. Training students to be comfortable with and knowledgeable about images is clearly important.

There are two complimentary approaches to training students in the video era. One is to include image-based examples in basic courses such as data structures (Sarkar and Goldgof 1998) and algorithms (Stevenson 2000). This provides a basic exposure to image computation for all students. The other is to provide courses in image computation for students who want to study the topic in more depth. One difficulty in creating new courses to teach image computation is the overlap with existing courses, for example computer graphics, computer vision and image processing. Unfortunately, these courses have many redundant components, and students do not have enough credit hours to take them all. Hence, the challenge is to present image computation in a course plan that properly exploits pre-requisites and does not duplicate material.

Colorado State University has introduced a course in image computation that is designed to follow and build off an established (and popular) course in computer graphics. The new course is centered on images: how they are generated, manipulated, matched, and symbolically described. It builds on students' knowledge of coordinate systems and the perspective projection pipeline. It covers some advanced topics in computer graphics not developed in the introductory course, most notably ray tracing. It introduces students to basic image processing concepts such as Fourier analysis. It then introduces students to basic computer vision topics such as principal components analysis, edge detection, and symbolic feature matching. The goal is to prepare students for advanced work in either computer vision or computer graphics, with an emphasis on elements common to both fields.

2. Course Design

There are many ways to design a course that integrates two or more aspects of image computation. At Colorado State, undergraduates spend three years working through the core curriculum in computer science before reaching a set of senior-level electives. One of these electives is the highly popular *CS410: Introduction to Computer Graphics*, which is offered every semester. Masters students can also take a limited number of 400 level courses for credit, and many choose the graphics course.

To meet our goal of introducing additional material on image computation into our curriculum, we initially considered broadening this course to include more image computation, or alternatively, offering a new companion senior-level course on image computation. However, both options had significant drawbacks. Modifying the introduction to graphics would require omitting material considered fundamental in an introductory graphics course. A new senior level course would compete with the existing graphics course for students, since most students would not have sufficient open elective slots to take both.

Our solution was to replace an existing masters level graphics course, *CS510: Computer Graphics*, with an image computation course. This solved several problems at once. First, since many of our better undergraduates were already taking CS510 in their last semester, interested undergraduates could still participate in this new course. Second, it provides an introduction to image computation for incoming graduate students planning on pursuing research in computer vision. Finally, and perhaps most interestingly, we found that the new course was able to retain some topics previously covered in our advanced graphics course. This is in keeping with our view that computer vision and computer graphics complement each other, and that increasingly, material required for understanding one is likewise required to understand the other.

We also obtained a considerable boost in what we could cover, since requiring the introduction to graphics course as a prerequisite meant our students are already familiar with many key topics. In particular, they are familiar with 1) coordinate systems including homogeneous coordinates, and coordinate system

transformations, 2) the pin-hole camera model, including perspective and orthographic projection, 3) simple 3D polygonal object modeling, and 4) scan conversion (Bresenham's algorithm) and region filling. Most importantly, their linear algebra skills have been honed through use in the introduction to computer graphics course, and the geometric interpretations of linear algebra. For example, they are familiar with the dot product of unit vectors being the cosine of the angle between the vectors.

The background of the students in terms of graphics is critical. Our graphics course emphasizes the mathematical underpinnings of graphics, rather than the use of graphical software tools. Students work their way through the mathematics of scan-line conversion, coordinate transformations, projection, and cubic curves (currently using (Hearn and Baker 1997) as their textbook). In the process, our students learn to intuit the connection between geometric concepts and linear algebraic expressions. Without this background, students could not succeed in our image computation course.

Our image computation course has four sections: 1) image generation, 2) image manipulation, 3) image matching, and 4) symbolic descriptions of images. Pedagogically, our idea was that the first section would make students intimately familiar with the physics of image formation, by discussing shading, ray tracing and radiosity, and having the students implement a ray tracer from scratch. The second section gets them to think of images as objects to be manipulated, not just looked at, and to think about the contents of images in a more formal manner. The major topics here are image transformations (affine, perspective and warping), and pixel interpolation. We would also like to cover image compression in this section, but so far time constraints and an already heavy workload have prevented this.

The third section addresses the question: "Do two images match?" The section begins with cross-correlation, and then goes on to principal components analysis and Fourier matching (with stereo and mosaicing as applications). The assignment for this section is to implement a PCA-based image retrieval system. The goal is to get students to think about the information present in images, and thus what operations make this information explicit. This leads naturally to the last section, which introduces the mainline computer vision topics of feature extraction and feature matching. It covers a subset of edge

extraction, point extraction, Hausdorff point matching, line extraction, and symbolic graph matching. The first time we taught the course, the project was to implement the line extraction algorithm of (Burns, Hanson et al. 1986); the second time the project was to implement a generalized Hough Transform (Ballard 1981).

As computer vision researchers, this general course outline sacrificed several topics that are near and dear to our hearts. "Mainline" computer vision topics are not introduced until the third and fourth sections, and pattern recognition, machine learning, computational geometry, and color constancy are among the topics not covered at all. Nonetheless, for students who do not intend to pursue research in computer vision, this course design gives them the familiarity they will need to understand, for example, image databases and/or biometrics. Just as important, for students who go on to study computer vision further, this course provides a background in image formation, image transformations, image matching, and symbolic descriptions of images. From this basis, they can pursue many research topics in computer vision.

3. Related Courses at Other Universities

To our knowledge, Colorado State University is the only university to introduce students to image computation through a prerequisite course in computer graphics. The University of Iowa uses a traditional course in digital image processing (course # 55:148) as the prerequisite for its course in image analysis and understanding (course # 55:247). The digital image processing course gives students a background in image digitization and restoration, region segmentation, edge extraction, and image compression prior to the computer vision course, and both courses are supported by a single textbook written by the instructors (Sonka, Hlavac et al. 1998). This sequence is probably better suited for a computer engineering department (as at Iowa) than a computer science department (as at Colorado State). Engineering students who choose to stop after one course would probably prefer image processing, while computer science students who take only a single image-related course are more likely to prefer a course in graphics.

Other universities have approached the problem the other way around, creating a new course in image computation that serves as a prerequisite for traditional courses in computer vision, graphics, or image processing. For example, the University of Rochester created a course called *Principles of Visual Computing* (CS290B) that is intended for sophomores and introduces students to simple 2D graphics (drawing lines and regions), alternate image representations (e.g. the frequency domain), and image transformations (texture mapping, warping). Carnegie-Mellon University has a course in *Image-based Modeling and Rendering* (15-869) which is intended for seniors and masters students and is divided into sections on 2D images (warping, mosaicing), depth images (z-buffer, interpolation, sprites) and 3D modeling (cameras, texture maps, geometry, radiance). This course is not integrated into an image computation sequence, however, in the sense that it does not have any prerequisites, nor does it serve as a prerequisite for other courses in graphics, vision, or image processing.

There are advantages and disadvantages to these approaches. By creating a course on image computation early in the curriculum (sophomore year), the University of Rochester has one course that can lay the framework for later studies in graphics, vision and image processing, allowing those subjects to be taught in depth at the undergraduate level. On the other hand, there are a limited number of courses that a student can take as an undergraduate, and introducing a sophomore course on image computation at CSU would require displacing one of the current core courses. It is unlikely that the general faculty would agree to this.

Carnegie-Mellon's approach avoids this problem by making image computation an elective senior/masters level course. In essence, they have added a new elective course to supplement existing courses in graphics, vision and image processing. This gives students as much flexibility as possible, in terms of which parts of image computation they want to study. Unfortunately, it also implies redundancy between the courses (since they are not prerequisites of each other).

Although many other schools offer courses in image computation, these examples illustrate the three basic approaches: as a core course early in the curriculum (Rochester); as a set of independent but overlapping courses (CMU); or by sequencing traditional courses to exploit the overlap (Iowa/CSU). At the University

of Iowa they are working within a computer engineering curriculum, so they use image processing as the prerequisite for computer vision. This paper describes our approach at Colorado State University, where we use computer graphics as the lead into image computation and computer vision.

4. From Theory to Practice: Spring 2000

The decision to convert the second semester graphics course into an image computation course was made in the summer of 1998, and was first taught by Ross Beveridge in the Spring of 1999. This initial course included some but not all of the design described above (the second section was missing, the third was less emphasized, and a module on 3D modeling was included). Based on that experience, the course was modified and then taught again in Spring 2000, this time by Bruce Draper. Readers should note that although we were fairly pleased with the Spring 2000 course, refinements continue.

Semesters at Colorado State University are 15 weeks long (plus one week for exams), and the course meets twice a week for 75 minutes. This gives us a total of 37.5 hours of instruction. Institutions with quarters or other semester schedules would have to adjust the syllabus accordingly. The lecture schedule for Spring 2000 – as it happened, not as it was planned – is shown in Table 1. All lectures are made available to students over the web, and can be accessed by readers at http://www.cs.colostate.edu/~cs510.

Week	Lecture	Topic	Source
1	Tuesday	Introduction	N/A
1	Thursday	Image Formation	T&V 2.1-2.4
2	Tuesday	Illumination & Reflectance	H&B 14.1-14.2
2	Thursday	Shading & Ray Tracing	H&B 14.6
3	Tuesday	Ray Tracing	H&B 14.6
3	Thursday	Ray Tracing	H&B 14.6
4	Tuesday	Ray Tracing & Radiosity	H&B 14.6-14.7

4	Thursday	Radiosity	H&B 14.7
5	Tuesday	Sampling	Lecture notes
5	Thursday	Image Transformations	Lecture notes
6	Tuesday	Color Space Transformations	H&B 15.1-15.10
6	Thursday	Morphing	H&B 16.5, notes
7	Tuesday	Triangulation	Lecture notes
7	Thursday	Interpolation	Lecture notes
8	Tuesday	Stereo	T&V 7.1-7.3
8	Thursday	Image Matching & Correlation	Lecture Notes
9	Tuesday	Eigenspaces	T&V 10.4
9	Thursday	Eigenspaces	T&V 10.4
10	Tuesday	Eigenspaces	T&V 10.4
10	Thursday	Fourier Matching	Lecture Notes
11	Tuesday	Fourier Matching	Lecture Notes
11	Thursday	Introduction to Features	T&V 4.1
12	Tuesday	Canny Edges & Corners	T&V 4.2-4.3
12	Thursday	Hough Line Extraction & Hausdorf Matching	T&V 5.1-5.2
13	Tuesday	Generalized Hough Transform	Lecture Notes
13	Thursday	Active Contours (snakes)	T&V 5.4
14	Tuesday	Matching with Interpretation Trees	T&V 10.1-10.2
14	Thursday	Matching with Pose: geometric hashing and geometric tree search	Lecture Notes
15	Tuesday	Matching with Pose (more)	Lecture Notes
15	Thursday	Summary & Student Feedback	N/A

Table 1: Schedule of Lectures for CS510: Image Computation. The dark lines indicate the boundaries between the four sections of the course. The last column gives the student's reference source for the material, with T&V being *Introductory Techniques for 3-D Computer Vision* (Trucco

and Verri 1998) and H&B being *Computer Graphics: C Version* (Hearn and Baker 1997). Course lectures can be accessed over the web at http://www.cs.colostate.edu/~cs510

To reinforce the major themes of the course, students do four programming assignments. In the Spring 2000 semester, the students built 1) a ray tracing program, 2) an image morphing program, 3) an image retrieval program based on principal components analysis, and 4) a generalized Hough transform matching program. This is a demanding set of assignments. We can assign this much work because 1) it is the second in a series of elective courses, implying self-selected and well-motivated students, and 2) the students are seniors and/or masters students who have experience writing large software systems. In another context with more typical undergraduates, we might try providing partial implementations of these projects and having the students complete them. However, we have not tried this in practice.

The most significant difference between the course as taught in Spring 1999 and in Spring 2000 was the replacement of a module on 3D modeling with the module on image transformations. This change was made for several reasons. In terms of the importance of image transformations, almost anyone now working routinely with imagery will encounter situations where both geometry and color representations are altered. Consequently, we felt it essential to introduce students to these concepts.

Although 3D modeling is central to both computer vision and computer graphics, from a practical standpoint we found we could shift some of this material to the pre-requisite graphics course, or omit it entirely. Specifically, we felt students need to understand the basics of 3D polygonal modeling, as well as parametric curves and parametric surfaces. Of these topics, the introduction to computer graphics course already covered polygonal modeling and parametric curves. By moving parametric surfaces, which are a natural extension of curves, into the introduction to computer graphics course, we ensure that students are exposed to this material. Finally, while in the past we taught constructive solid geometry, we felt that the image transformation material was of greater practical importance.

5. Student Reaction

The student's overall reaction to this syllabus and course was that they liked the material, but some thought the workload was excessive. This conclusion is based largely on informal student comments. Another source of information is the anonymous student survey collected at the end of every course at Colorado State University. The survey contains 22 statements about the course, classroom, teacher and student, and the students are asked to reply to each statement on a scale of zero (strongly disagree) to four (strongly agree). Table 2 summarizes the responses to the most relevant statements. Eleven out of twelve students strongly agreed that the course was intellectually challenging, and that they put considerable effort into it. All at least agreed that the course was good, with seven of twelve strongly agreeing with this statement. Unfortunately, the survey does not ask directly about the workload, although the fact that eleven of twelve strongly agreed that the course was intellectually challenging might be viewed as support for the informal observation that the students remarked about the workload.

The most poorly worded question in the course survey asks: "The text and/or course materials were appropriate?". This was the only question to elicit neutral responses, and it would be interesting to know if the criticism was of the text or the course material. Our belief, based on informal student comments as well as the positive responses to other questions on the survey, was that the neutral responses were in regard to the textbooks. We used two textbooks for this course, Hearn and Baker's *Computer Graphics: C Version* (Hearn and Baker 1997) and Trucco and Verri's *Introductory Techniques for 3-D Computer Vision* (Trucco and Verri 1998). The rationale was that the students already owned Hearn and Baker, since it is required for the CS410 prerequisite, and that Trucco and Verri was the only textbook that covered both principal components analysis and feature extraction/matching (Maxwell 1998; Maxwell 2000). Through these two texts, the students had reference materials for the first, third and fourth segments of the course. Unfortunately, they did not have a required reference for the section on image manipulation. Although this material is covered in many image processing texts, most students relied on the lecture notes for this part of the course and were dissatisfied. Another problem was that principal components analysis was the topic of three lectures and one programming assignment, but is only five pages in Trucco and Verri's text. PCA is a

complex subject, and the students wanted a more thorough reference. After the spring semester was over, Wendy Yambor completed a Master's Degree with us, and a better tutorial introduction to PCA algorithms, complete with MATLAB code, is included in her thesis (Yambor 2000) and webpage (see http://www.cs.colostate.edu/~vision/html/projects/yambor/thesis.htm.) This material is being used as the basis for the PCA lectures in Spring, 2001, and is the only significant refinement between the Spring 2000 and Spring 2001 versions of the course.

6. Conclusions

In terms of subject material, using image computation as a bridge between traditional computer graphics and advanced computer vision works well. The student's knowledge of coordinate systems, the perspective projection pipeline, and related material allows them to probe image formation, manipulation, matching and description more deeply then would otherwise be possible, and having graphics as a prerequisite eliminates the need for redundancy between the two courses. The biggest drawback is that there is no single textbook that adequately covers the material in this image computation course.

Statement	**Strongly Agree**	**Agree**	**Neutral**	**Disagree**	**Strongly Disagree**
The course was intellectually challenging	11	1	0	0	0
The assignments increased my understanding	11	1	0	0	0
Class sessions increased my understanding	7	5	0	0	0
The text and/or course materials were appropriate	3	5	4	0	0
I put considerable effort into this course	11	0	1	0	0
Overall, I would rate this course as good	7	5	0	0	0

Table 2: student responses to course survey.

References

Ballard, D. (1981). “Generalizing the Hough Transform to Detect Arbitrary Shapes.” Pattern Recognition **13**(2): 11-122.

Burns, J. B., A. R. Hanson, et al. (1986). “Extracting Straight Lines.” IEEE Transactions on Pattern Analysis and Machine Intelligence **8**(4): 425-455.

Hearn, D. and M. P. Baker (1997). Computer Graphics: C Version. Upper Saddle River, NJ, Prentice Hall.

Maxwell, B. A. (1998). “Teaching Computer Vision to Computer Scientists: Issues and a Comparative Textbook Review.” International Journal of Pattern Recognition and Artificial Intelligence **12**(8): 1035-1051.

Maxwell, B. A. (2000). A surevy of compuer vision education and text resources. Workshop on Undergraduate Education and Image Computation, Hilton Head, SC.

Sarkar, S. and D. Goldgof (1998). “Integrating Image Computation in Undergraduate Level Data-Structure Education.” International Journal of Pattern Recognition and Artificial Intelligence **12**(8): 1071-1080.

Sonka, M., V. Hlavac, et al. (1998). Image Processing, Analysis, and Machine Vision. Pacific Grove, CA, PWS Publishing.

Stevenson, D. E. (2000). Computational Geometry and Image Processing Applications for an Undergraduate Algorithms Course. Workshop on Undergraduate Education and Image Computation, Hilton Head, S.C.

Trucco, E. and A. Verri (1998). Introductory Techniques for 3-D Computer Vision. Saddle River, NJ, Prentice-Hall.

Yambor, W. (2000). Analysis of PCA-Based and Fisher Discriminant-Based Image Recognition Algorithms. Department of Computer Science. Fort Collins, CO, Colorado State University.