Review of Computer Vision Education

George Bebis, *Member, IEEE*, Dwight Egbert, *Senior Member, IEEE*, and Mubarak Shah, *Fellow, IEEE*

***Abstract*—Computer vision is becoming a mainstream subject of study in computer science and engineering. With the rapid explosion of multimedia and the extensive use of video and image-based communications over the World Wide Web, there is a strong demand for educating students to become knowledgeable in computer imaging and vision. The purpose of this paper is to review the status of computer vision education today.**

***Index Terms*—Computer vision, education, image processing, research, teaching.**

I. Introduction

VISION is perhaps the most important of the human senses. It provides us, seemingly effortlessly, with a detailed three-dimensional (3-D) description of a complex and rapidly changing world. Computer vision, the study of enabling computers to understand and interpret visual information from static images and video sequences, emerged in the late 1950s and early 1960s and is expanding rapidly throughout the world. It belongs to the broader field of image-related computation and relates to areas such as image processing, robot vision, medical imaging, image databases, pattern recognition, computer graphics, and virtual reality.

During the past ten years, computer vision has grown from a research area to a widely accepted technology, capable of providing a dramatic increase in productivity and improving living standards. The key factors that have contributed to this increase are the exponential growth of processor speed and memory capacity. Computer vision is a source of powerful tools for industry and other disciplines, including multimedia, robotics, manufacturing, medicine, and remote sensing. The potential practical benefits of computer vision systems are immense. It is anticipated that computer vision systems will soon become commonplace and that vision technology will be applied across a broad range of products, revolutionizing our lives.

Computer vision is becoming a mainstream subject of study in computer science and engineering. With the rapid explosion of multimedia and the extensive use of video and image-based communications over the World Wide Web, every student in computer science and engineering should receive some basic education related to computation with images [1], [2]. It has even been argued that images represent a very important data structure that needs to be covered in every book on data structures [2]. Currently, there is strong industry demand for computer vision scientists and engineers well versed in this technology—people who understand computer vision technology and know how to apply it in real-world problems. As a result, many students are expected to pursue careers in computer vision and related areas in the future. This fact has triggered many discussions within the computer vision community during the last few years to address important issues related to computer vision education [1]–[5]. In 1996, a panel discussion was held at the Institute of Electrical and Electronics Engineers (IEEE) Computer Vision and Pattern Recognition Conference (CVPR) on improving teaching of image-related computation [2]. In the last five years, three IEEE workshops on computer vision education issues have been organized, in conjunction with CVPR, to address important educational issues [3]–[5].

Although computer vision is becoming a very important and practical area in computer science and engineering, it is not very well represented in course offerings at most institutions [2]. In order for institutions to respond to future demands for computer vision professionals, they need to broaden and improve their curriculum by including computer vision and related image computing courses in their undergraduate and graduate programs.

The purpose of this paper is to review recent efforts to integrate computer vision in the curriculum successfully and effectively. In general, this integration seems to serve three main purposes. The first is to teach students concepts in computer vision and related areas, such as image processing and robotics. The second purpose is to use computer vision as an enabling technology to improve teaching and knowledge acquisition. The last, and probably most challenging, purpose is to use computer vision as a vehicle for integrating teaching with research. The emphasis of this review is on computer vision education; however, the discussion addresses closely related areas in certain cases, such as image processing.

II. Traditional Computer Vision Courses

Traditionally, computer vision is offered at the upper-division undergraduate or graduate level in most institutions. In some cases, it is offered as a complement or continuation of an image-processing course, and in other cases, as a stand-alone elective. This section reviews several important issues related to traditional computer vision courses.

A. Prerequisites

Computer vision has special technical requirements that places it at an advanced level with several prerequisites, such as data structures, image processing, calculus, linear algebra, and numerical methods. Students who take the computer vision

Manuscript received May 14, 2002; revised September 3, 2002. This work was supported by the National Science Foundation under CRCD Grant 0088086.

G. Bebis and D. Egbert are with the Computer Vision Laboratory, Department of Computer Science, University of Nevada—Reno, Reno, NV 89557 USA.

M. Shah is with the Computer Vision Laboratory, School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816 USA.

Digital Object Identifier 10.1109/TE.2002.808280

course are usually expected to understand and integrate knowledge from various disciplines—something that is rarely true. Planning for students, especially computer science students, who may not possess a broad background is very important for making the course a rewarding experience for them. In general, there are two approaches to handle this issue. First, students can be required to have taken a number of prerequisite courses before taking computer vision. This approach, however, limits significantly the number of students, especially undergraduates, who can take the computer vision course. The second and more challenging approach is to evaluate carefully the general knowledge that the students are expected to possess by the time they take computer vision and design the course around that knowledge to enable them to gain a strong understanding of the fundamentals of computer vision. Maxwell [6], for example, has argued in favor of presenting concepts as algorithmically as possible to computer science majors. A possible disadvantage of this approach is that the students might not get exposed to important details about how certain techniques work. An interesting discussion about other approaches can be found in [7]. Also, a challenging approach to cover gaps in background knowledge "using students to teach students" is presented in Section III-C.

B. Content

Computer vision research aims to understand the representations and processes underlying vision in sufficient detail so that they may be implemented on a computer. There are two main motivations for this: 1) to develop computer vision systems as a method of testing and evaluating models of human or other biological vision systems and 2) to use engineering approaches to design and build computer systems to solve practical problems. Depending on the background and interests of the instructor(s) offering the course, the emphasis can be shifted toward either direction.

The first direction is usually found in courses offered mostly by programs in cognitive science or psychology. Units of computer vision taught within artificial intelligence (AI) courses also have this "biologically inspired" flavor. The second and probably more popular direction is found within programs in engineering and computer science. Of course, there exist "hybrid" courses that emphasize both directions, such as "Image Representations for Vision," a course offered at the Massachusetts Institute of Technology (MIT). Hybrid computer vision courses, however, are less common mainly because most instructors do not have a background both in human perception and computer vision.

Deciding what to teach in a computer vision course is a topic of debate. Because of its breadth, it is impossible to cover all topics within a semester. Also, the dynamic research nature of the computer vision field produces new material every year, some of which needs to be integrated into current computer vision courses. Selecting which topics and techniques to cover is not commonly agreed on among computer vision educators [8]. Many times, depth is sacrificed for breadth. As a result of its interdisciplinary nature, computer vision appears in the curriculum of various departments. Obviously, the interests, aptitudes, and learning requirements of students in various departments vary considerably, as do the backgrounds and motivations of those teaching computer vision courses. Therefore, the content of the course needs to be tailored to the current interests and strengths of the students, as well as to those of the instructor.

A recent survey [8] has considered approaches to computer vision courses as taught at various institutions. The results of the survey reveal that computer vision courses offered at various institutions can be classified into the following five groups:

1) classic image processing;
2) classic computer vision;
3) application oriented;
4) focused;
5) high-level.

The first category represents traditional image-processing courses with limited emphasis on computer vision. Courses falling in the second category are comprehensive courses attempting to provide breadth of knowledge by offering a survey on a variety of topics. Most of these courses cover such topics as edge- and region-based segmentation, feature extraction and representation, camera geometry, calibration, stereo, motion, texture, shading, and object recognition. Courses in the third category emphasize the application aspect of computer vision and pay much less attention to theory. The emphasis in the focused-oriented category was on presenting the material by having different foci, such as object recognition and content-based image retrieval. The last category emphasizes advanced research-oriented computer vision topics.[1]

III. Innovative Computer Vision Courses

In this section, a number of innovative computer vision courses are reviewed. They are referred to as innovative in the sense that they attempt to improve the effectiveness of teaching and learning by exploiting students' background knowledge, using interactive technology or using nontraditional teaching pedagogy.

A. Computer Vision Courses Exploiting Students' Background Knowledge

The courses described in this section assume that students have backgrounds in certain areas to improve the effectiveness of teaching and learning. An example is a computer vision course designed to follow and build upon a computer graphics course at Colorado State University [7]. Their motivation was teaching the computer vision course in a way that exploits prerequisites and does not duplicate material. Specifically, the course capitalized on students' prior knowledge of various computer graphics topics, such as coordinate systems and the perspective pipeline. The material covered in the course included computer vision topics, such as image generation, processing, matching, symbolic description, and advanced computer graphics topics such as ray tracing. In this respect, the two courses complemented each other very well (i.e., material required for understanding one was likewise required to understand the other). A drawback with the structure of this course is that it is not a stand-alone course. On the positive side,

[1]More information about the courses considered in the survey is available from http://www.palantir.swarthmore.edu/~maxwell/visionCourses.htm

the students were better prepared to take the computer vision course, and the instructors were able to cover more material than usual.

At Rey Juan Carlos University in Spain [9], standard algorithmic techniques were integrated in an undergraduate image-processing/computer vision course to enable students to better understand image operations and algorithms. The emphasis was on algorithm design techniques for image processing, that is, teaching the students how to design efficient (both in terms of time and memory) image-processing algorithms, an issue of practical importance. A complementary strategy where computer vision examples were integrated in an algorithms course to improve the understanding of algorithms is discussed in Section V-A. The course at Rey Juan Carlos University covered several traditional topics, with each topic being related to a number of algorithms and major design techniques (e.g., quad-trees, median filtering, and split-and-merge were related to the divide-and-conquer technique). The lectures stressed the efficiency aspects of image-processing algorithms and were reinforced through a number of laboratory assignments and practical projects. In some cases, students were given an algorithm and were asked to identify the design technique that best described it. In other cases, a simple image task was given to them, and they were asked to devise an algorithm for that task using a particular design technique. Overall, the course improved the students' understanding of various image-processing/computer vision algorithms while reinforcing their knowledge of algorithm design techniques.

B. Computer Vision Courses Using Interactive Technology

Active involvement of students in the learning process allows them to learn more effectively and successfully. Traditionally, computer vision courses are taught by a series of lectures, laboratory sessions, and assignments. Although the subject matter of computer vision is inherently visual, theoretical concepts and algorithms are discussed with few demonstrations and exercises. Krotkov from Carnegie Mellon University, Pittsburgh, PA, said in his statement at the 1996 CVPR panel [2], "As every computer vision scientist knows, it is difficult to understand how a particular algorithm works without actually applying it on various images, using various parameters. It is hard to convey this experience to the students using only a few images. Even more difficult is trying to demonstrate to the students how algorithms that operate on video sequences perform." Obviously, the traditional approach hinders the learning process significantly. Ideally, students should be able to understand the algorithms, their properties, and the theory behind them by testing them on different inputs and by changing parameter values. Essentially, laboratory sessions and assignments have this role. However, few algorithms are assigned for implementation because of time constraints, and it is difficult to maintain synchrony between class lectures and lab sessions. To alleviate these problems, several computer vision educators have considered using interactive technology to offer "hands-on" learning experiences to the students.

A common approach to improving student learning is through creating online HTML-based materials containing multimedia objects, such as embedded images, sounds, and demonstrations [20]–[22]. Usually, a software package such as Khoros [23] is used in conjunction with the HTML material. This approach is not fully interactive and relies on platform-dependent software. Recently, several computer vision educators have built fully interactive materials by integrating software written in Java into HTML documents [24]–[26]. Java has the advantage of being machine independent and is freely available. An interesting idea to make computer vision algorithms publicly available over the Web using Java is also proposed in [27]. Many times, computer vision researchers make their source code available to the community (e.g., through "anonymous file transfer protocol"); however, it is not always easy to compile and run the code because of platform compatibility issues. The idea in [27] is to build interactive Java-client/common gateway interface (CGI)-server applications to enable testing computer vision algorithms over the Web. Implementing computer vision algorithms on a machine-independent platform (e.g., using Java) and making them available to the community would be very beneficial when teaching these algorithms in computer vision courses.

Besides building online materials and online "demos," the learning experience can be further improved using "true" collaborative student learning. The University of Iowa, Iowa City, has achieved this objective using a state-of-the-art electronic classroom [28]. All students are provided with a fully functional networked workstation that serves as a tool for educational material as well as for direct experimentation. Each lecture is a combination of presentation of concepts, examples, and practical exploratory problems. The instructor's station can send live video to all other workstations in the classroom. The video can come from a charge-coupled device (CCD) camera, a visual presenter, or a VCR. Using special software, any X-Window can be shared between the instructor and the student screens, allowing independent student groups to be formed. The lecture is delivered via an instructor-controlled and shared Netscape window. An additional window is shared with students to allow two-way interaction between the instructor and the students. Examples and exploratory problems are implemented using Khoros [23]. After the theoretical part of a topic is presented, the instructor could present an example to the students, or the students could work on an exploratory problem (e.g., test the Canny edge detector on a number of images and experiment with its parameters). The instructors have found that the biggest advantage of lecturing in an electronic classroom is the incorporation of practical problems within the lecture, allowing the students to achieve basic understanding of important concepts during the lecture in a collaborative way.

While the concept of course delivery in an electronic classroom is very attractive and effective, cost constraints do not allow many institutions to adopt this approach. Using interactive course materials, however, is a more viable approach that can be adopted more easily. Section VII presents a more extensive review on software tools and environments that are available for teaching and research purposes.

C. Computer Vision Courses Using Student-Centered Learning

Traditionally, computer vision is taught based on lectures and textbooks. Although this approach can be effective for certain kinds of learning, it is a poor method for meeting higher cognitive objectives, such as critical thinking, and does not motivate

students to learn [10], [11]. In this section, several innovative approaches to teaching computer vision are reviewed. The innovative element in these courses is in the teaching pedagogy used. Specifically, all the courses reviewed below have several common goals: to improve student learning, help students develop problem-solving skills, expose them to recent research results and the process of doing research, and instill appropriate habits in them, such as lifelong learning. Lifelong learning, in particular, is very important in the context of a rapidly changing field such as that of computer vision. A draft report of the joint IEEE/Association for Computing Machinery (IEEE/ACM) task force on Year 2001 Model Curricula for Computing [12] mentions that "Fundamentally, teaching students to cope with change requires instilling in those students an attitude of mind that promotes continued study throughout a career."

To meet these objectives, several computer vision educators have adopted a number of innovative approaches based on nontraditional educational methods that target student-centered learning. The two most common forms of this approach are *problem-based* learning and *seminar-based* learning. Seminar-based approaches focus on group discussion and have less emphasis on implementation. Problem-based approaches, on the other hand, emphasize research and implementation using group work.

The undergraduate computer vision course at California State University, Hayward, [13] follows a problem-based approach by integrating experimental-based learning experiences (ELEs) [14], [15]. The main goal was to transfer effectively to the students both important skills and knowledge. Following the definition given in [13] of an experimental learning experience being the presentation of a problem or task with no solution given, students were required to do "research" in order to solve them. That is, they defined the problem, reviewed the literature, broke down the problem, investigated potential solutions, implemented, tested, and documented the results. The emphasis was not only on imparting to students computer vision knowledge but also on instilling in them investigative skills and appropriate habits that would allow them to tackle unsolved computer vision problems. The course material included some traditional topics from image processing and computer vision (e.g., pixel-based and area-based operations, segmentation, and recognition) and was structured around the ELEs. Three ELE examples are given in [13]: the green-screen experiment (i.e., a green-screen system that filters out the green-screen backdrop and inserts any image in the background); the pseudocoloring experiment (i.e., using a grayscale image of a woman's hair, obtained from a special X-ray scattering machine, to pseudocolor certain concentric rings toward the outer part of the hair that might indicate the presence of cancer); and the dog-tracker experiment (i.e., detecting in a scene the motion of a dog). In addition to the ELEs, traditional programming assignments were given to the students. Two interesting aspects of the course are: 1) the projects involved both software development and vision hardware system configuration and 2) emphasis was placed on group work and collaborative learning using corporate organizational models. Both of these important issues have been brought up by a number of industry researchers in the 1996 CVPR panel [2].[2]

Very similar objectives were defined for a computer vision course taught at Sheffield Hallam University, South Yorkshire, U.K. [16]. In contrast to traditional computer vision courses, where the goal is to familiarize the students with a variety of techniques, the goal of this course was to instill appropriate habits in the students. The course targeted real-world computer vision applications (e.g., in manufacturing), and the objective was to teach students how to evaluate and identify appropriate prior knowledge in order to design successful machine vision systems. Through a number of inquiry-based group assignments, the students learned about lighting techniques, cameras, lenses, digital image processing, and interfacing. At the end of the course, students had a better understanding of the strengths and weaknesses of computer vision technology.

A seminar-based approach was compared with a problem-based approach in an undergraduate computer vision course at the University of Otago, Otago, New Zealand [17]. Both approaches were found to be more effective than the traditional approach based on lectures. Between the two approaches, the problem-based approach was found to be more effective. In the seminar-based approach, a particular computer vision topic was discussed every week. The students were required to read in advance a "classic" paper in the field and to find and review a recent, related journal paper on their own. The review was based on a number of questions similar to those found in journal paper review forms. They were also expected to work on a computer vision project chosen on their own and submit a report at the end. The seminar-based approach achieved a number of important objectives, such as teaching the students the essentials of computer vision, showing them how to search the literature to find new techniques, and helping them to develop critical-thinking skills. The main weakness of the seminar-based approach was that the students were not able to achieve an in-depth understanding of the techniques discussed because of time constraints and high-level discussions of the topics.

The problem-based approach attempts to alleviate some of the problems associated with the seminar-based approach by being more focused and offering more hands-on experiences to the students. The course was divided into six units, including the following topics: reconstruction and filtering, edge detection, shape representation, object recognition, stereo correspondence, and optical flow. A classic paper or a textbook chapter was assigned for reading before each session. Students were also required to review a recent related paper as in the seminar-based approach. For each unit, the students had to implement a classic technique and submit a report. The choice of implementing a classic technique rather than a recent one was made to leave more time for experimenting with the technique and analyzing the results. Overall, the problem-based approach was found to be more effective than the seminar-based approach, despite the fact that fewer topics were covered. Concentrating on a few techniques while trying to teach the skills of thinking, analysis, and evaluation seems more important than teaching more classic techniques.[3]

Traditionally, computer vision is taught in bottom-up fashion; that is, students are taught the fundamentals (i.e., background knowledge) before being exposed to primary material. Recently,

[2]More information about this course is available from http://classes.monterey.edu/CST/CST332–01/world/index.htm

[3]Further information is available from http://www.cs.otago.ac.nz/research/vision/Research/TeachingVision/teachingvision.html

the computer vision course at the University of Otago was taught in a top-down fashion, based on the idea of "using students to teach students" [18], [19]. The approach was problem-based again; however, the students were exposed to material that required knowledge they did not previously have. To cover their deficiencies in background knowledge, they had to work backward to teach themselves what they needed to know. This scenario was inspired from the way active researchers fill their own gaps in background knowledge. In particular, the students reviewed recent computer vision papers and chose a set of background topics with the help of the instructors. Each student then worked on one background topic and prepared a written tutorial for use by the others. Each student received peer reviews of their tutorials from the other students and the instructors. This "background skills" unit was implemented in a six-week segment that was part of a two-semester computer vision course. The rest of the course followed the problem-based approach. Compared with previous years, students performed better in the course when the top-down approach was used. Overall, acquiring background knowledge by following a top-down approach has the benefit of preparing students to become more active participants in their learning and encourages them to follow a research career. Although it is difficult to draw any general conclusions about the effectiveness of this approach based only on the results of this small pilot study, teaching background knowledge in a top-down fashion is an interesting approach that warrants further investigation.

IV. Complementary Computer Vision Courses

In this section, a number of complementary computer vision courses are reviewed. These courses are not based on traditional computer vision principles and tools, and their role is to support and enhance the computer vision curriculum.

A. Advanced Computer Vision Topics

Computer vision education needs to adapt rapidly to the pace of modern innovations. Recent advances in the field have made imperative the development of new and improved computer vision courses at both undergraduate and graduate level. In particular, computer vision education has begun to follow computer vision research into interdisciplinary areas, such as virtual reality, human–computer interfaces, and medical imaging. In this section, several representative computer vision courses are reviewed.

An elective undergraduate course called "Digital Video Special Effects" has been offered twice at the Georgia Institute of Technology, Atlanta, with great success for the last two years [29]. This course is not a pure computer vision course, but rather, a combination of computer vision, image processing, computer graphics, digital video, digital audio, and cinematography. The goal of the course was to teach students video and audio manipulation, processing, analysis, and synthesis. The course contained technical lectures on the subject of digital video processing and was built upon the students' computer science and engineering core background, offering them the opportunity to apply and extend their knowledge in the context of a very motivating application. The students were expected to assimilate the theoretical background on digital video and apply it to generate their own video production.

Coupling the technical aspects of the course with the entertainment aspects of generating special effects has made this course very appealing to the students. The course had two projects: the first was to familiarize the students with the subject on an individual basis, while the second emphasized group work and was much more demanding. The projects involved both computer vision and computer graphics concepts (e.g., tracking colored objects and replacing them with renderings). The students were also required to do a critical evaluation of a segment from a movie. Using a similar approach to "students teaching students" (see Section III-C), they were required to study an effect of their choice and then try to explain to the class, based on their own understanding, how the effect was generated.

The course is a good example of an application area that relies on the combination of knowledge from several other areas. It provides an important opportunity for students to integrate and use their acquired knowledge to solve practical problems. Despite the obvious benefits, adopting such a course in other institutions would be rather difficult because of the heavy software, hardware, and equipment requirements as well as the need for significant lab space.[4]

The "Visual Interface" course offered at Columbia University, New York, NY, is an elective undergraduate course [30]. The goal of the course is to introduce students to the use of visual input, not only for data but also for the control of computer systems. Using imagery as a direct form of control input represents a major research direction within the computer vision field. The course content is based on conference and journal articles, doctorate dissertations, and online Web demonstrations. The lectures are organized around the following six case studies of working visual systems:

1) visual gesture interpretation;
2) visual information retrieval;
3) natural language interpretation;
4) visual guidance of vehicles;
5) visual surveillance;
6) visual methods for biometrics.

The emphasis is on surveying and analyzing the architecture, algorithms, and underlying assumptions of these systems but also on exploring these systems' foundation in cognitive science and artificial intelligence. Comparisons among different designs and decisions are also emphasized. No prior knowledge of computer vision is required. In fact, the case studies cover many traditional computer vision fundamentals and techniques, but not in the traditional order or depth.

The students were required to finish two programming assignments and a final paper or project. The first programming assignment was to design a "visual combination lock" (i.e., to process a sequence of images to determine if the user had placed some body part, such as his or her hands, in a predetermined sequence of locations). The second assignment was to parse an image into English. Both assignments contained a creativity step

[4]More information about the course and a sample of projects are available from http://www.cc.gatech.edu/classes/cs4803d_99_spring/

(i.e., something extra the students had to do on their own to extend the capabilities of the system). The final paper was required to be a state-of-the-art report of some aspect of visual interfaces, while the final project was an implementation of a small-scale system that could process visual human data for a particular application (e.g., visual burglar alarm or a visual-based computer log-in).

Adopting this course in other institutions is straightforward. It does not have any special software requirements [an improved version of Sun's X-Windows Imaging Library (XIL) is used at Columbia University] and can be tailored to the specific interests of the instructor without difficulty.

At the University of Iowa, a two-course sequence on medical imaging is offered, called "Physics and Analysis of Medical Images" (I & II), [28]. The first course covers the physics of medical imaging modalities [e.g., X-ray, computed topography (CT), position emission tomography (PET)] and the image-processing techniques used to analyze the images. Specific topics include medical image reconstruction, enhancement, analysis, and clinical interpretation. The second course concentrates mostly on nonionizing radiation modalities (e.g., MRI and ultrasound) and advanced topics of medical image analysis, such as geometric operations, noise removal, classifying and recognizing objects, and 3-D visualization. The course largely targets students in biomedical engineering, electrical and computer engineering, medicine, and radiology. Students who take the course are expected to have backgrounds in linear systems analysis, signal analysis, college physics, college biology, and human physiology.

Lectures are conducted in an electronic classroom following the lecture–example–experiment teaching philosophy (see Section III-B). Students were required to complete several programming assignments (e.g., use region growing to find the borders of the left ventricular chamber) and a semester project, which required some type of quantitative assessment of a disease process identified with a medical image and using an automatic segmentation method.

B. Mathematical Methods for Computer Vision

Computer vision is a broad-based field of computer science that requires students to have both a good and a broad mathematical background. Most students, however, especially undergraduates, do not usually have sufficient mathematical background. In an effort to make up for students' weak backgrounds, most educators either spend too much time on teaching background concepts or instead skip the mathematical details and proceed immediately to demonstrations and implementation. Both approaches have their strengths and weaknesses. The first approach allows for an in-depth coverage of a relatively small number of computer vision topics, while the second approach favors covering a broader range of topics but in much less detail. In the 1996 CVPR panel [2], a general agreement was that the math/theory side of computer vision should not be sacrificed in favor of fancy demonstrations that hide the details of how the techniques work.

In order to satisfy this goal, instructors are forced to spend a significant amount of time teaching background concepts. To handle this problem and establish a strong foundation for enabling student research in computer vision, an elective course on mathematical methods for computer vision has begun appearing in the curriculum of several institutions. At Rensselaer Polytechnic Institute (RPI), Troy, NY, Stewart and Mundy co-instructed a course "Mathematical Techniques in Computer Vision" in Fall 2000.[5] At Stanford University, Standford, CA, Tomasi has taught several times over the last few years a course "Mathematical Methods for Robotics and Vision.[6] At the University of Nevada—Reno (UNR), Reno, Bebis taught a course "Mathematical Methods for Computer Vision" in Fall 2001.[7] Related courses have been taught at the Hebrew University of Jerusalem, Jerusalem, Israel (i.e., "Math for Computer Vision and Robotics"[8]), the University of Zagreb, Coratia (i.e., "Mathematical Methods in Image Processing[9]), and the University of Lund, Lund, Sweden (i.e., "Mathematical Methods in Computer Vision and Image Analysis"). Similar courses in related areas have also been developed (e.g., "Mathematical Methods for Computer Graphics" at Stanford University). The subject seems to have reached a certain level of maturity in some of these areas, justified by the availability of several books [31], [32].

Choosing the structure, prerequisites, content, and assignments for such a course is not easy. The most challenging issues are what to assume is known and what to teach. Depending on the student audience (e.g., undergraduates or graduates), the number and type of prerequisites can vary. Because of the interdisciplinary nature of the field, a broad range of mathematical techniques and tools are being used in computer vision research. Clearly, the traditional engineering background is not sufficient. Even active researchers find it challenging to keep track of and understand all of these techniques and tools. The lack of textbooks makes this task even more difficult. Most computer vision textbooks assume that the students already know how these techniques work or provide a brief description in the appendix. In most cases, however, the students do not have the background assumed, and they cannot understand a technique based on the material provided in the appendix.

The courses mentioned previously were primarily taught based on lecture notes, book chapters, and research papers. From the Web pages of those courses, there is a bias seen in the choice of topics, depending on the background and interests of the individual instructor. The "Mathematical Methods for Robotics and Vision" course at Stanford University is offered at the undergraduate level and is probably the most mature among all the courses mentioned. It has been taught several times in the past, and Tomasi has written some excellent notes, which are available from the course's Web page (see footnote 6 at the bottom of the page). The course has a theoretical flavor. Homework emphasizes mathematical problems, and programming assignments are rather limited. The goal of the course is to teach students some of the key tools of applied mathematics without really emphasizing computer vision or

[5]http://www.cs.rpi.edu/~stewart/math_vision/

[6]http://www.stanford.edu/class/cs205/

[7]http://www.cs.unr.edu/~bebis/MathMethods

[8]http://www.cs.huji.ac.il/course/mfvr/

[9]http://ipg.zesoi.fer.hr/teach/mmip/index_en.html

robotics. The prerequisites of the course are calculus and linear algebra. The particular subjects covered in the course include algebraic linear systems, function optimization, ordinary differential linear systems, statistical estimation, tensor fields of several variables, partial differential equations, and sparse linear systems.

The course at RPI has been taught only once so far. The course was offered at the senior undergraduate level and had calculus and linear algebra as prerequisites. No background in computer vision was assumed. The topics that the instructors chose to teach were linear algebra (review), analytical and projective geometry, transformations, statistics, estimation, robust estimation, and numerical techniques. In contrast to the Stanford University course, the RPI course emphasized some computer vision topics, such as projective geometry, cameras, multiview geometry, and application in face and object recognition. Homework problems were mostly mathematical, and there were two programming assignments on estimating transformations.

At UNR, the course was taught for the first time in Fall 2001 and subsequently in Spring 2002. The course was offered at the junior/senior undergraduate and introductory graduate levels. The prerequisites were data structures, calculus, linear algebra, and probability and statistics. No background in image processing or computer vision was assumed, other than some basic knowledge of image manipulation. To ensure the selection of a representative set of topics, the following directions were explored:

1) searching the computer vision literature to summarize the major mathematical tools used in computer vision research;
2) considering the mathematical background sections provided in the appendix of most computer vision and related textbooks;
3) collecting information from computer vision course Web pages.

The topics included in the Fall 2001 syllabus were: image-processing transformations (e.g., Fourier transform and wavelet transform), linear algebra review, dimensionality reduction (e.g., principal component analysis), probability and statistics review, classification (e.g., Bayes' classifier, linear discriminant analysis, neural networks, support vector machines, Bayesian nets, and hidden-Markov models), clustering, calculus review, optimization (e.g., simulated-annealing, genetic algorithms, and constrained optimization), numerical methods (e.g., singular value decomposition, Newton's method, and gradient descent), estimation (e.g., least-squares, robust estimation, and Kalman filter), projective geometry, computational geometry, and algorithms (e.g., hashing, greedy algorithms, dynamic programming).

This list of topics provides enough material to be covered in a two-semester course sequence. Unavoidably, the Fall 2001 offering covered fewer topics than initially planned and focused mostly on pattern recognition techniques. Because of time constraints, emphasis was placed on teaching the students "what is possible" and "how to learn details" when necessary, rather than covering everything in detail. When teaching students a number of mathematical techniques, the instructors must teach them not only how to apply each technique but also how to recognize what mathematical approach to apply to new situations. Discussing math in the context of practical applications is a good way to address some of these issues. The course at UNR emphasized both theory and applications in contrast to the Stanford University and RPI courses that were taught mostly in a theoretical manner.

Course lectures were organized as self-contained modules. Each module was accompanied by lecture notes, a list of reading material, case studies, and pointers to software resources. Each time a math method was covered in class (e.g., principal components analysis), a number of case studies (e.g., face recognition, object recognition, and estimation of high-dimensional probability distributions) were also discussed to make the theory clear and more interesting. There were neither homework assignments nor formal exams, but there was a short quiz after the completion of each topic. Also, there were two programming assignments, one comparing principal components analysis (PCA) with linear discriminant analysis (LDA) for face recognition and one using wavelet features for fast image retrieval. The students were also required to choose one of the mathematical methods discussed in class, study three or four papers discussing applications of this technique in computer vision, summarize the papers, and give a short presentation to the rest of the class.

C. Robotics Courses

Depending on the institution, computer vision can be used in support of robotics, or computer vision can be the primary focus, with robotics as an example application. For example, Gini at the University of Minnesota, Twin Cities, MN, describes learning computer science through robotics projects, and computer vision is used as a navigational tool [34]. Clement describes an instructional laboratory at the U.S. Naval Academy, Annapolis, MD, designed for teaching both robotics and machine vision courses, which gives equal weight to each of the two topics [35]. Some areas of emphasis, such as manufacturing technology, seem to integrate computer vision and robotics so closely that they become one field (robot vision) [36], [37]. Krotkov [38] describes the use of robotics as a teaching tool throughout the curriculum in several specialized courses at Carnegie Mellon, including computer vision, AI, and mechatronics. The use of robotics has become widespread, beginning with simple LEGO systems for use in K–12 and freshman engineering courses as described by Wang [39]. This elementary use of robotics usually does not include computer vision applications. However, as robotics is incorporated throughout the computer science and engineering curriculum, computer vision applications begin to become a standard part of robotics. Murphy [40] describes five outcomes of a robotics competition course, and one of them is "competence with common industrial vision and machine perception techniques." Murphy also writes an introduction to a special issue of IEEE INTELLIGENT SYSTEMS on robotics and education [41]. This issue has four articles that describe a range of issues, from setting up a robotics lab to integrating robotics with other topics [42]–[45].

D. Other Supportive Courses

Several supportive computer vision courses exist in the curricula of many institutions. Among them, the most common ones are image processing, computer graphics, pattern recognition, and AI. Image processing can be considered as low-level computer vision and is usually a prerequisite for many computer vision courses. Computer graphics can be thought of as the inverse of computer vision, since the objective is the synthesis of an image, rather than analysis. There is significant overlap between graphics and vision (e.g., cameras and illumination models). Quite often, vision concepts are discussed in graphics courses, as graphics are discussed in vision courses (e.g., insert artificial objects in real scenes). A computer vision course requiring computer graphics background knowledge was presented in Section III-A. Pattern recognition, and particularly statistical pattern recognition, courses cover a number of techniques that have great application in computer vision. Finally, AI courses usually include an overview of the most important computer vision areas.

V. Using Computer Vision to Improve Learning and Knowledge Acquisition

One of the challenges computer science educators are faced with today is motivating students to learn. An invaluable tool to be used in this process is assigning interesting programming projects that impart cohesiveness to the course concepts and generate a lot of enthusiasm for learning. Quite often, however, students are discouraged by the nature of the projects they are asked to implement because the projects lack connection with the real world. Assigning challenging programming projects that connect what the students are learning to real applications can capture their minds and motivate them to learn more.

Computer vision is an excellent tool for demonstrating general principles in computer science and engineering, as well as for improving learning by increasing student motivation. Because the visual experience has strong applications appeal, combined with strong intuitive appeal, computer vision is ideal to use in studying programming, data structures, algorithms, and hardware-related courses. It can help impart cohesiveness to the course concepts and bring them closer to real life than is the usual practice. Its interdisciplinary nature also makes computer vision an excellent learning tool for teaching students to integrate and use their acquired knowledge.

Integrating computer vision examples in core courses also has the advantage of exposing a large number of undergraduates at an early stage to computer vision and related areas. It has been argued several times that all computer science students need to be exposed to the fundamentals of image-related computation to prepare them for today's demanding computing environment [2]. Despite the increasing importance of image manipulation in everyday computing, few undergraduates get exposed to this area, because most departments offer computer vision at the advanced undergraduate or introductory graduate level. Ideally, adding a computer vision course with less prerequisites at a lower level would solve this problem. However, most departments cannot afford adding more courses into their curriculum. Integrating computer vision examples into core courses seems to be the next most feasible solution.

Computer vision examples can be integrated in various courses naturally without detracting from the original teaching goals. Instructors can easily adopt these examples in similar or related courses while keeping the original topics in each course mostly intact. Students usually need some assistance with learning how to read, write, and display images, but no other knowledge of image manipulation is required or assumed. In most cases, a skeleton program having the previously described basic functionality is provided to the students to save time.

A. Using Computer Vision Examples in Core Courses

1) Programming: Programming assignments should demonstrate and reinforce important programming concepts and techniques discussed in class. Usually, however, programming courses lack assignments that are sufficiently engaging. They specifically emphasize programming skills and fail to expose the students to a broad range of computational problems. Nonmajors, in particular, often perceive these courses as difficult, because they emphasize computer systems rather than applications, and programming style rather than hands-on exploration. Several educators have tried to make these courses more interesting by including selected topics from more advanced courses [46]. Demonstrating programming concepts using image manipulation concepts can be accomplished very naturally. Arithmetic and logical operators, for example, can be demonstrated by applying them on images. The demonstration will certainly be more impressive and interesting than using almost any other data. Looping structures can be demonstrated using topics such as image sampling, quantization, or scaling. Finally, -if- structures can be demonstrated using image thresholding.

There are several interesting image-related projects that have been introduced into various programming courses. Two projects, one based on edge enhancement and the other on image compression, were used in CS1/CS2 courses at the Polytechnic University of Madrid, Madrid, Spain, to demonstrate loops, one-dimensional (1-D) and two-dimensional arrays, files, recursion, and trees [47]. Contrast enhancement, histogram equalization, and hidden message extraction from images were used in a CS1 course at Northeastern University, Boston, MA, to demonstrate data formats, file organization, and file manipulation and also to help students practice loops, decision statements, assignment statements, arithmetic expressions, and arrays [48]. The students in that course found it very exciting that the images they were using in their assignments were images of Mars collected by the National Aeronautics and Space Administration Viking Orbiter. At Duke University, Durham, NC [49], the final programming assignment in a CS1 course was based on image manipulation. Simple image compression, based on run-length coding, restoration using median filtering, and image expansion, were used to reinforce control constructs and arrays.[10] In a freshman engineering honors programming course at Purdue University [50], West Lafayette, IN, students

[10]See http://www.cs.duke.edu/csed

worked in four-person teams on an image analysis project that involved halftoning, using histogram thresholding, and edge detection, using the Laplacian mask. The objectives were to reinforce programming concepts, group work, and the ability to handle large, open-ended problems. The project seemed to be rather ambitious; however, the overwhelming voice heard from the students was that they developed incredible amounts of programming skills.

Computer vision concepts were also used in a C++ programming course at Michigan State University (MSU), East Lansing, MI, that was designed especially for nonmajor graduate students in science and engineering [51]. The course was designed for mature students who wanted to learn programming for their research, and the course capitalized upon their good math background and research interests to motivate them to learn C++ using computer vision concepts. Three of the programming assignments given to the students were computer vision related. The first was based on connected components and invariant object recognition, the second on image compression and motion detection, and the last one on ray tracing, camera models, and reflection models. Lecture time was spent on topics related to the projects, such as image enhancement, masks, convolution, camera models, perspective projection, and reflection models. Probably, too much material was covered within a short period of time; however, the course was offered to mature graduate students. Programming concepts emphasized by projects included data representation, file input/output, nested for-loops, 2-D arrays, recursion, and class reusability.[11]

In another course offered at the Harvard University Extension School, Cambridge, MA [52], case studies from some practical computer science application areas, including image processing and computer vision, were used to teach introductory applied computing to general-education students. The emphasis was on problem-solving and design techniques, rather than programming, thus addressing some of the issues mentioned in the beginning of this section. In each case study, the students were first introduced to the subject and then asked to use previously implemented systems to perform design and problem-solving tasks. The studies related to computer vision were based on image enhancement and face recognition. Adobe Photoshop was used to demonstrate simple image enhancement techniques, while a simple face recognition algorithm, written by the authors, was used to demonstrate various issues involved in the design of a face recognition system, such as choice of an image-similarity metric and sensitivity to lighting, facial expression, and head pose.

Computer vision examples have also been integrated in the "Introduction to Computer Science" (CS 201) course at UNR. This typical CSAB CS1 course is the first programming course for most students. The introduction of computer vision in this course comes at the end of the semester with two lectures and an image-processing term project [53]. One lecture covers research principles and computer vision basics, followed by a second lecture with questions about the project. Image data are introduced as an example of 2-D arrays. Basic concepts, such as edges in images being changes in brightness, are explained so that students can understand the processing they will perform. The project consists of a menu-driven image-processing program with several functions, including file read and write, negative, rotate, threshold, and basic filter. The necessary programming does not involve anything beyond what was covered in the course. Students are given a function to read a black-and-white portable gray map (PGM) file and use this function as a model on which to base the writing of their own file-write function. All other functions simply involve array manipulations. One of the image-processing functions is an operation of the students' own choice. Many students were very creative in their choice of operations, and most students enjoyed the project. The visual feedback made them feel they could really accomplish much with only one programming course.

To develop the programming skills of radiography undergraduates, Allan and Zylinski [54] have used image-processing examples in a programming course at Royal Melbourne Institute of Technology, Melbourne, Australia. In this course, students learn to program in Visual Basic by implementing a digital subtraction angiography package.

2) Data Structures: Solving a problem efficiently requires knowledge of efficient algorithmic techniques. Implementing the solution efficiently, however, requires knowledge of efficient data structures. Introducing data structures to students through computer vision examples can provide a very clear demonstration of these issues in the context of a real application. Computer vision deals with algorithms that process and extract important information from images. Knowledge of the theoretical basis of these algorithms is not sufficient, because one has to implement these algorithms efficiently using appropriate data structures. Many computer vision tasks offer a natural way to introduce basic data structures, such as arrays, queues, stacks, lists, trees, and graphs.

The data structures course at the University of South Florida (USF), Tampa, FL, has been taught several times using computer vision examples [55]. Students taking the course are not required to have any prior knowledge in image processing or analysis. To make up for the students' lack of background in image-related computation, one lecture is dedicated to discussing the fundamentals of image generation, representation, and processing. Seven programming assignments are described in [55], each one emphasizing a different data structure. Students work in groups to complete the assignments. The goal of the first assignment is to help students practice arrays by implementing simple image-processing operations, such as thresholding and binary morphology. In the second programming assignment, the students practice stacks and queues by implementing the connected components algorithm. The third assignment teaches students to design and manipulate lists. Lists are demonstrated by using them to construct the run-length code of an image or to store the connected components. Trees are demonstrated in the fourth assignment, where the students construct quad-tree representations of binary images. The fifth assignment introduces graphs

[11]The course material and assignments are available from http://www.cse.msu.edu/~cse891

through the task of image-based spatial reasoning. Hashing is demonstrated in the sixth assignment in the context of a 2-D shape-classification problem. Finally, median filtering is used to demonstrate sorting in the last assignment. The assignments were designed to emphasize good software principles, such as modularity, information hiding, and reuse. A quantitative analysis of the effectiveness of these assignments using pre- and post-assignment tests is reported in [56]. The results of the analysis indicate that the assignments have contributed significantly to the understanding of data structures and basic software design principles.[12]

A similar approach has been used to integrate computer vision examples in the data structures course at UNR. The main difference from the USF approach is that the assignments have been augmented with inquiry-based material to motivate the students to extend and improve the algorithms to be implemented. The goal was to improve the students' critical thinking skills and give them a taste of what research is all about. To motivate them, extra credit was offered for interesting ideas. Students were not expected to have any prior knowledge of image processing or manipulation. In the beginning of the semester, they were introduced to the fundamentals of computer vision and image processing with two lectures. The first lecture emphasized the area of computer vision and discussed promising applications, accompanied by video-based demonstrations to spark their interest. The second lecture was more technical and discussed image generation, representation, and manipulation. Four programming assignments were given to the students. The goal of the first assignment was to help them practice dynamically allocated arrays, constructors, destructors, copy-constructors, and operator overloading. The objective of the assignment was to implement an image-processing package having the capability to perform some simple image-processing operations, including addition, subtraction, negation, translation, scaling, and rotation. Examples of inquiry-based questions include implementing scaling more effectively (i.e., the basic scaling function they are asked to implement is based on subsampling or upsampling) or eliminating the holes from the rotated image (i.e., the basic rotation function they are asked to implement is based on the forward transformation). Only grayscale images are used in the first assignment. The goal of the second assignment was to help them improve their skills in using templates. The objective was to extend the capabilities of the image-processing package so that it can handle both grayscale and color images. In the third and fourth assignments, the students were asked to implement a simple system to recognize U.S. coins from images. The third assignment illustrated stacks, queues, recursion, and running times, while the fourth assignment emphasized lists. Examples of inquiry-based questions in this case included how to use the histogram of the image to automate thresholding or how to improve the performance of coin recognition. Computer-vision-based assignments on trees and graphs are currently under development (e.g., using decision trees for object recognition and graphs for region-adjacency representation).[13]

3) Algorithms: Courses on the analysis of algorithms often include little programming and do not emphasize practical applications. Many times, students perceive this course as rather impractical because of its heavy theoretical content. Students lack motivation to study theory unless they are exposed to interesting applications. At least two efforts have been documented to integrate image-related computation concepts into algorithms [57], [58]. In both cases, the students demonstrated higher motivation and better understanding and retention of the course material.

In [57], the emphasis was on the implementation of specific algorithms and data structures, empirical analysis of their performance, and comparisons with their theoretical time complexity. Four programming assignments were designed to allow the students to acquire hands-on experience with algorithms and grasp difficult concepts. Students were assumed to have no prior experience with images. The first assignment was on representation of images. The purpose was mainly to familiarize the students with image encoding; however, they also implemented some simple functions (i.e., compute mean gray-level value), measured the running time of each function, and determined the dependency of the time on the image size. The second assignment was on median filtering, which requires fast sorting algorithms. The students implemented and compared various sorting techniques by varying the window size in median filtering. In the third assignment, they implemented connected components, using graph operations and disjoint-set structures. The objective was to compare the performance of connected components using a linked-list representation of disjoint sets and a disjoint-set forest. The last assignment was on image compression using Huffman codes. The purpose was to familiarize students with greedy algorithms. The running time and compression rate of the implemented compression algorithm was measured and compared using Unix's *gzip* utility.[14]

The emphasis in [58] was on algorithm design strategies (e.g., greedy and divide-and-conquer) and advanced programming techniques (e.g., object-oriented programming and graphical user interfaces). The course had five assignments, and the students had to work individually to complete each assignment. Most of the programming assignments in this course were related to computational geometry, rather than to computer vision; however, the program assignments are included, since there is a relation between the two areas. Students were assumed to have no prior knowledge in computational geometry or computer vision. The purpose of the first assignment was to introduce the students to time complexity. This introduction was accomplished using various algorithms for the computation of the convex hull. By comparing the running times of various algorithms for the computation of the convex hull (e.g., brute force, MergeHull, and QuickHull),

[12]More information about the assignments is available from http://marathon.csee.usf.edu/~sarkar/ds_ip.html

[13]More information is available from http://www.cs.unr.edu/~bebis/CS308

[14]The course material is available from http://www.csee.usf.edu/~eugene/algs/

the students came to understand practical issues related to algorithm complexity. The purpose of the second assignment was to demonstrate the divide-and-conquer strategy. A fast algorithm for convex-hull computation, called QuickHull, was used for this purpose. A greedy triangulation algorithm was used to demonstrate the greedy strategy in the third assignment. In the fourth assignment, the topic of approximation algorithms was covered using the convex-hull problem again. In the last assignment, parallel algorithms and the concepts of threads and synchronization were introduced by implementing a parallel convolution filter.

Overall, image-related computation examples seem to be a perfect fit for use in algorithms courses. Complexity issues can be demonstrated rather successfully using images, because of the large amount of data that need to be processed. In a related course [9], concepts from algorithms are used to improve the understanding of image operations (see Section III-A).

4) Hardware-Related Courses: Few efforts have arisen to integrate computer vision concepts in computer vision courses. One of them is the Microprocessor Engineering (CS/EE 336) course at UNR, which introduces the details of microprocessor architecture, assembly language, and interfacing to an embedded single-board computer (SBC). As part of the interface programming experience, the students must learn about command and data transfer protocols and several file formats. The concepts of computer vision and image processing were easily incorporated through the analogy of sound (which has been covered previously) as a 1-D signal compared with an image as a 2-D signal. The computer vision lecture module for this course includes a review of the scientific research method and introduces computer vision concepts in the context of the course. For example, a digital camera is one instance of an embedded system with a microprocessor controller and a menu interface. The camera menu is compared with the SBC menu written by the students. The computer vision laboratory session is used to reinforce student understanding of the serial interface hardware, the software command protocol, and the file formats, which have always been included as part of this course. The lab is structured to demonstrate the research process by having the students "discover" the transfer protocol between the computer and camera from experimental data with the help of a Web document, which describes how others are also learning the protocol through experimenting. This experiment is directly related to the course's study of SBC interfacing as well as providing an exciting application, which allows the students to understand that digital images are simply streams of data.

Computer vision concepts have also been integrated in an embedded computing course [59]. Today's modern embedded computers are more sophisticated and can process still images, video, and audio. As a result, there is a clear need in embedded computing courses to cover all these technologies that go into modern embedded computing devices. The adoption of vision-related technologies in modern consumer products makes embedded computing courses a good fit for computer vision examples. At Clemson University [59], Clemson, SC, the emphasis of the embedded computing course is on teaching students how to construct multimedia embedded computing devices. Among other things, the students also learn how to construct and program hardware to operate as a digital video camera. Framegrabbing (i.e., a user-level program to interact with the kernel and device driver), image display (i.e., code to display an image in a window on the monitor), and compression (i.e., programs to implement standard codecs, such as LZW, RLE, and Huffman) are some of the lab assignments related to this project.

There is a strong feeling among computer vision researchers, especially those working in industry, that computer vision students need laboratory work and project work with a systems approach [1]. Students need to acquaint themselves with computer vision hardware, such as cameras, lenses, and lighting. They also need to become aware of issues related to real-time processing and hardware tools. Including more material or labs in traditional computer vision courses or adding new courses to the curriculum are not feasible approaches. On the other hand, adding computer vision modules in hardware-related courses might be the best way to address this problem.

5) Math-Related Courses: Traditionally, mathematics is taught at the undergraduate level in classes on calculus, differential equations, linear algebra, etc. The material consists largely of formal definitions, theorems and proofs, and some examples. Most students feel the material to be too dry, which cause them to lack motivation. The instructors teaching mathematics courses always wish that the students were more motivated and interested to learn the material. Mathematics is also addressed in courses in physics, engineering, and computer science. In these courses, mathematics is essentially applied to solve some engineering and physics problems. The students are assumed to have a background in mathematics in order to succeed in these courses. The instructors teaching engineering and physics courses always wish that the students had a more solid background in mathematics.

At the University of Central Florida (UCF), Orlando, an honors course "Computer Vision Guided Tour of Mathematics"[15] is offered to reinvigorate interest in mathematics among students and to reverse a decline in enrollment in mathematics-related courses. To address these issues, an innovative approach to teaching mathematics is used by illustrating concepts, without going into formal proofs, and by discussing real-world computer vision applications. The computer technology and some fine software packages, such as MATLAB, make this approach possible. This course consists of three modules: face recognition, motion estimation, and object recognition. Each module starts with a discussion of a real-world problem. First, the students are introduced to these real-world problems and asked to do simple surveys on the Web and make short presentations on what they learned about the problem (e.g., face recognition). The computer vision solution to the problem is discussed at an algorithmic level. Students are next introduced to key mathematical concepts related to the solution of this problem (e.g., eigenvector, eigenvalues, matrix inversion, determinant, and principal component analysis).

[15]See http://www.math.ucf.edu/~xl/vision02/syl02.htm

Finally, students are assigned to implement the solution to the problem, using MATLAB, and demonstrate it in the class.

B. Using Computer Vision Examples in Other Courses

Computer vision examples have been integrated in various courses throughout the science and engineering curriculum. Schultz at the University of North Dakota, Grand Forks, ND [60], [61] uses image processing as a visualization tool in a signal processing (SP) course. SP courses are usually taught within electrical engineering departments and primarily cover 1-D signal theory. According to Schultz, when various SP algorithms are applied to real-life signals, such as speech or music, students have a hard time interpreting the results because these signals cannot be visualized easily. Using 2-D signals (i.e., images), however, can help the students to understand important mathematical concepts, such as convolution, space–frequency duality, sampling, reconstruction, and restoration, by first observing the results and then delving into the underlying theory of these concepts. At Drexel University, Philadelphia, PA, Cohen and Petropulu [62] have also integrated image computation examples into a digital signal processing (DSP) lab course.

In [63], a digital camera and a computer imaging program were used to teach students in college-level science courses the physics of light. In particular, the spectral responses of some common materials in the visible spectrum, such as iron, water, air, soil, and vegetation, were demonstrated using imaging to help students understand that color variation in natural materials is the result of how visible light is emitted or reflected.

In [64], image processing was used to enrich the teaching of a course on structural design for buildings. A key objective was to enhance the visual critical analysis skills of the students by teaching them how to apply hypotheses based on structural principles to explain phenomena and features visible in images of structures. The use of image processing to enhance, annotate, and manipulate images was crucial in conveying structural concepts to the students.

In [65], computer imaging was used to improve engineering design associated with soil and water engineering. The main goal of engineering design courses is to demonstrate to the students applications that relate engineering judgment to real-world problems. Using imaging, engineering judgment can be incorporated with realistic situations. For example, using cut-and-paste tools, imaging can help to illustrate how installation of practices and structures might change the appearance of an area without expensive construction or time constraints. Imaging allows for different structures to be placed in the same area in order for the students to see which appears to be the most effective to the area. This process demonstrates to the students that the best solution obtained using mathematics may not always be the most appropriate solution in real life.

C. Using Computer Vision Examples to Teach High-School Students

The motivation for introducing computer vision and image processing into the high-school (and even middle- and primary-school) curriculum is vividly described by Thomas *et al.* [66], as "Vision is the sense through and by which we perceive and understand our world. … Learned eye–body coordination makes it possible for us to act and/or react smoothly and efficiently in all sorts of vision-guided situations. … It is also a powerful medium for communicating complex scientific ideas, especially those involving scientific processes. … We have never seen a technology so appealing to students of all ages as scientific visualization."

Thomas describes visiting a one-room elementary school in Montana and showing students images of Mars taken from the Viking spacecraft. The students became involved in how to use the images to answer questions about Mars, such as the size of craters and characteristics of other geologic features. He reports that "Over the next hour, the class took first an interest, then ownership, then pride in their investigation of Mars." Thomas describes a practical image-processing system for use in the classroom and introduces the National Institutes of Health (NIH) Image software, which is available free [67]. This system can be used by any teacher to bring scientific visualization to the classroom. Thomas also provides several suggested lesson plans for using the system. For example, he describes how high-school teachers can use the NIH Image histogram function with National Oceanographic and Atmospheric Administration (NOAA) thermal images to examine ocean temperature distributions.

A group of publications by Greenberg, *et al.* [69]–[78], spanning almost ten years, describes a similar experience with using image processing for upper-elementary and secondary teaching. Their project "Image Processing for Teaching" (IPT) started as a response to the 1988 solicitation by the National Science Foundation (NSF) for Projects to Promote the Effective Use of Technology in the Teaching of Science and Mathematics. Part of the motivation for the IPT project derives from a 1983 survey of teachers in which nearly 85% of the respondents noted that their preferred style for both teaching and learning was "visual." The IPT project also uses the NIH Image software and distributes a huge database of images, including 20 000 images from the Voyager spacecraft. Greenberg points out that "…these images are not just pretty pictures. They are the original data sets from recent research in a wide variety of disciplines." In addition to this database of images being an effective teaching tool, Greenberg points out that students, while learning, have an opportunity for original scientific discovery, because most of the images have not been thoroughly examined by professional staff: "The scientific community does not have the person-power to fully explore all these images. For example, the 20 000 images in the Voyager spacecraft data set … have been only partially explored by researchers."

Greenberg and others have observed that image processing is basically a mathematical operation, and even though students may not start out thinking about mathematics, after some experience with image processing, they start to think about the underlying mathematical operations. In fact, Tanimoto [74]–[77] has used image processing as the basis for teaching mathematics in his "Project on Mathematics Experiences Through Image Pro-

cessing," or MEDTIP. Tanimoto points out that "The fact that digital images are represented by arrays of numbers attests to the relevance of mathematics in the exciting realm of visual imagery." The project offers students alternative experiences in "arithmetic, algebra, problem solving, geometric transformations, and digital representation."

Software was developed for the project, which ranges from a simple "pixel calculator" through contrast enhancement and image warping to convolution. Tanimoto found that students did indeed respond to the excitement of image processing and learning mathematics. In addition, he also found that the work satisfied another student "need to work with a computer using something other than a video game or word processor." In addition to basic image processing, Stockman [78] has developed a computer vision teaching module for use with high-school seniors through college seniors. This module starts with defining how a digital image is captured and the meaning of the array of numbers. The first computer vision application is to write an algorithm for counting the number of holes in a solid object, such as a structural beam, using a binary image. The algorithm examines the image two rows and two columns at a time to count "interior corners, I" (3-hole pixels) and "exterior corners, E" (3–nonhole pixels). The total number of holes then is calculated by $(E - I)/4$. The module then continues with other applications, including a maze search and connected component analysis. In order to cover meaningful concepts and tasks in a short period of time, Stockman uses carefully chosen algorithms, simple fixed images in ASCII file format, and simple single-task programs.

VI. Using Computer Vision to Integrate Teaching With Research

There has been a strong movement lately to involve students in research [11], [79]. Immersing students into research is a key factor for enhancing their critical thinking, creativity, self-confidence, and ability to work on collaborative projects and for motivating them to pursue graduate studies and engage in lifelong learning. Current science education standards [80], however, reflect increased emphasis on teaching science "facts" and decreased emphasis on teaching how those facts are generated, how they are used to understand complex systems, and how they may be applied to new and complex situations. Exposing students to real data and scientific inquiry experiences can help them realize that science is a process, instead of a collection of facts to memorize. Student research is an invaluable tool to be used in this process. Making research an integral part of the curriculum and embedding research pedagogy within the curriculum will offer a more balanced and effective educational experience.

Computer vision is an ideal area for integrating research with teaching. It has an immediate appeal to most students because of their intimate relationship to visual experience. Students can literally "see" the results of applying theory to solve practical problems. Its interdisciplinary nature can assist students in attaining a higher level of competence in science, mathematics, engineering, and technology areas, issues that have been raised by a recent advisory committee to the NSF [79]. Finally, computer vision is an excellent learning tool for teaching students to integrate and use their acquired knowledge, while at the same time providing a high level of motivation.

A. Traditional Methods for Integrating Teaching With Research

The three most popular approaches for exposing students to computer vision research are

1) joining faculty research teams;
2) taking elective courses and independent studies;
3) getting involved in summer research.

These approaches have demonstrated success; however, they suffer from several drawbacks. The most serious drawback of the first approach is that it targets a rather small number of students. The problem with the second approach is that many students may not have the opportunity to take the elective courses or independent studies as a result of constraints imposed by their degree programs. The last approach tries to alleviate these problems through intensive summer seminars; however, it still targets a rather small student population. Another serious problem is that these approaches lack sufficient organization and planning of activities. Involvement in research projects and independent studies are ad hoc in nature, and these opportunities can change significantly from semester to semester. The summer research programs are more established and consistent; however, they only occur during the summer. Educators who have been involved in these activities. such as Research Experiences for Undergraduates (REU), agree that a research experience that lasts only a few months in the summer does not allow sufficient time to complete a serious project [81], [82].

B. Integrating Research Into Computer Vision Courses

In a fast developing field such as computer vision, education and research should not be independent from each other. The major challenge for computer vision educators, however, is how to cover the basic theory while, at the same time, exposing the students to state-of-the-art research in the field. Usually, research results are integrated in advanced graduate-level computer vision courses through lectures and assignments. In this case, the integration can be done with relative success and usually leads to student publications [8]. Integrating research results in introductory computer vision courses is usually not an option because of time constraints, or it happens only as part of a final project. Despite the difficulties, there exist several documented efforts to integrate research results in introductory computer vision courses using a student-centered learning approach [9], [13], [16]. Two specific examples of this approach (i.e., seminar-based and problem-based) were discussed in Section III-C.

C. Integrating Research Into Core Courses

Integrating research results in elective courses cannot reach a large student population. The obvious solution to this problem is integrating research results in a number of core courses, thereby giving every student the opportunity to get

to know what research is all about. The learning approach employed in these core courses is more inquiry-based than research-based. The fundamental difference between inquiry- and research-based learning is the prior state of knowledge of the broader community. In research, it is unknown by all; in inquiry, it is only unknown by the learner. Thus, it can be safely argued that from a student perspective, pursuit of inquiry-based learning should be functionally equivalent to conducting research. Within a computer science and engineering curriculum, computer vision research results can be integrated relatively easy into a number of core courses, such as programming, data structures, and algorithms. Several such attempts have been discussed in Section V-A.

D. NSF Programs Supporting Integration of Teaching With Research

Student research and the integration of research into educational activities has been a topic of funding interest to many agencies. The NSF, in particular, has always been encouraging innovative educational development. During the last few years, fostering integration of research and education has become a principle strategy in support of NSF's goals. Most NSF programs today require a strong educational plan with the research plan. The most notable example is the NSF CAREER program. Other NSF programs supporting student research are the Research Experiences for Undergraduates (REU) program, the Educational Innovation (EI) program, and the Combined Research-Curriculum Development (CRCD) program. Currently, there are two REU sites offering research experiences in computer vision, one at UCF[16] and one at the University of Michigan-Dearborn. A few CRCD programs with computer vision components have been funded in the past, one at Columbia University [37] and one at the University of Notre Dame, Notre Dame, IN. Currently, there is one CRCD program with computer vision focus at UNR.[17]

E. An Overview of the REU Program in Computer Vision at UCF

The computer vision lab at UCF has served as a national site for REU in computer vision for the last 15 years. This lab has been supported by a series of grants from the NSF, totaling 1.6 million dollars. There are 150 undergraduates from 12 institutions throughout the country who have participated in the program; half of the undergraduates have gone to graduate schools; undergraduates have coauthored 60 papers; seven undergraduates have written honors in the major theses; six students are now faculty members in different universities; and five students have started their own companies. The key elements of this REU model are

1) experience for a calender year to allow time to complete a substantial project;
2) assignment of a faculty advisor from his or her own school to each participant;
3) immersion of the participants in research;
4) follow-through over the year.

The project starts in the summer. The first major activity for the students is the short course in computer vision. Since the students do not have prior background in computer vision, this course quickly introduces them to the subject. The topics covered include imaging geometry, edge detection, region segmentation, 2-D shape, stereo and shape from shading, and motion. While some of the same core topics are covered every year, some topics also change from year to year based on the current emphasis within the research groups. After the short course, the students are introduced to several possible research problems by the faculty; the students are also invited to discover their own problems to research. After a few days of discussion, each student is assigned a research problem. To begin, students will read some research papers, record some video sequences, and implement some known algorithms. The faculty meets with all students in a group twice a week (Tuesdays and Fridays), during which time each student is asked to present a report about his or her project. Half of the students report on Tuesday, and the other half on Fridays. The research continues throughout the summer. The students from other institutions return to their respective schools and continue their research during fall and spring with their mentors. At the end of the academic year, students are asked to write a report about their project. Students who are successful in getting some interesting results are encouraged to submit papers to conferences for publication. From their experience, the investigators have found two main deficiencies in the students' backgrounds relative to preparation for research in computer vision: mathematics and programming. In this connection, the students are directed to read selected research papers and understand their contents, particularly the math and the implementation of algorithms. More details are provided in [82].

F. An Overview of the CRCD Program in Computer Vision at UNR

The CRCD program at UNR started officially in January 2001. The overall goal of this project is to integrate the results of recent and ongoing research in computer vision into the computer science and engineering curriculum. In contrast to more common approaches that propose integration at the senior level through the offering of advanced courses or research projects, the UNR model attempts to achieve integration of teaching with research at all levels, leading to a comprehensive instructional program, offering systematic and constant research experiences for as many students as possible. The project seeks to accomplish this goal by immersing students into research through systematic and structured activities starting at the freshman year and continuing until graduation and graduate school, making research an integral part of each student's education.

A key idea of the UNR approach is "injecting" research results into core courses throughout the curriculum. This approach

[16]see http://www.cs.ucf.edu/~vision/reu-web/REU.html

[17]see http://www.cs.unr.edu/CRCD

forms the "skeleton" of the UNR model, around which more traditional approaches are integrated. Computer vision research results are being incorporated into one freshman-level (i.e., introductory programming), one sophomore-level (i.e., digital design), and two junior-level core courses (i.e., data structures and microprocessors, see Section V-A). Second, to set a strong foundation for enabling student research in computer vision, a new junior-level course in mathematical methods for computer vision is being developed (see Section IV-A). Current and ongoing research results are being integrated into this course to make it another research experience. A new senior- and introductory graduate-level course is also being developed based on state-of-the-art research results in the area of object recognition. Finally, a "distributed" model of summer research experiences for undergraduate and graduate students is being implemented. During the summer, students (junior, senior, and graduate level) have the opportunity to do research at various research laboratories all over the country. Last summer, ten students participated in the summer program and participated in research at various sites, including the UCF, the Los Alamos National Lab, Lawrence Livermore National Lab, Honeywell, IBM Almaden, Ford, and International Game Technology.[18]

VII. Computer Vision Instructional and Software Resources

The following review is by no means exhaustive; its main purpose is to give an idea of some instructional and software tools currently available for teaching and research purposes. An extensive list of software tools is available from http://www-2.cs.cmu.edu/afs/cs/project/cil/ftp/html/vision.html, and a review and comparison of several computer vision software packages can be found in [25]. Instructional resources for teaching in computer vision fall into two broad categories: 1) software and teaching tools developed by other educators or researchers and made available for general use and 2) lessons, experiments, and software extensions based on commercial products.

In the first category, some of the tools are rather special purpose but might be of interest to anyone teaching computer vision. Khuri *et al.* [84] have developed a set of stand-alone programs that demonstrate the principles of different image compression algorithms. Programs to demonstrate run length encoding, quadtree compression of bit map images, and JPEG are available at http://www.mathcs.sjsu.edu/faculty/khuri/publications.html. Another useful tool, by Biancardi *et al.* [85], is for creating computer vision tutorials and is called TuLIP (Tutorial for Learning Image Processing), which is a language based on Pacco, which is derived from Tcl/Tk. TuLIP operates in two modes: in teacher-mode, one can create computer vision tutorials, and in user-mode students can learn and practice image-processing and analysis techniques. For courses in which students do their own programming, the use of class libraries in C++ programming environments is described by Roman [86]. Also, a class library called CLIP is described by Robinson [87], which can be used to implement image-processing tasks in small programs.

For courses in which extensive software support is desired, there are several image-processing packages that have been developed for computer vision and image-processing instruction. Intel, for example, has recently released two very powerful software packages called IPL and OpenCV. IPL supports image-processing algorithm development, while OpenCV supports computer vision algorithm development. IPL is available for Windows only, while OpenCV is available both for Windows and Linux. Both packages are publicly available from http://www.intel.com/software/products/opensource/.

Depiero [88] describes his SIPTOOL (Signal and Image Processing Tool), which is a multimedia software environment for demonstrating and developing signal and image-processing techniques. SIPTOOL is shareware and can be downloaded from http://www.ee.calpoly.edu/~fdepiero/csip_tool/csip_tool.html. It is intended to be used both for in-class demonstrations and for student programming projects. An image-processing package originally called JVision (later NeatVision), developed in Java by Paul Whelan at Dublin City University, Dublin, Ireland, is described by Braggins [89]. Whelan uses NeatVision in his own computer vision course and makes it available as shareware from http://www.neatvision.com/.

Another Java complete computer vision package called JVT (Java Vision Toolkit) is described by Powel, *et al.* [25]. JVT is offered under the general Gnu public license at http://marathon.csee.usf.edu/~mpowell/jvt/index.htm. For courses using some of the standard mathematics programming packages, computer vision tools are available as supplements to the packages. Eddins *et al.* [90] describe their experience with the MATLAB Image Processing Toolbox. Likewise, Jankowski [91] describes the development of courseware materials to be used with Mathematica.

For institutions with the necessary resources, several commercial image-processing packages are available. Sohi *et al.* [92] describe their use of AVS Express, by Advanced Visual Systems, in a digital image-processing course. AVS Express is a package for general scientific visualization applications using a visual programming environment and can be used in computer vision and many other scientific fields. Hanna [93] describes the development of a computer-aided learning (CAL) program for teaching image processing written in Visual Basic and C, with links to another commercial package, Global LAB Image. Global LAB Image by Data Translation [94] is described as "a complete, customizable imaging software package, ideal for scientific and general-purpose imaging applications." Finally, several articles describe the use of KHOROS software by Khoral, Inc., in teaching computer vision and image-processing courses [21], [22], [95]. Khoral [23] describes its software as offering "a sophisticated visual programming environment with access to hundreds of data processing and visualization tools, as well as a complete software development environment for extending the system for a particular application domain."

[18]More information about the UNR-CRCD program is available from http://www.cs.unr.edu/CRCD

VIII. Recommendations

The responsibility of computer vision educators is to take appropriate action to advance the level of computer vision education. Kevin Bowyer's statement in the 1996 CVPR panel [2] carries a clear message: "Those of us who specialize in image-related computation should take the lead and develop new and interesting ways of educating students with/about image-related computations." A number of recommendations are given hereafter with the goal of stimulating creative thinking and encouraging more educators to reconsider how computer vision education is delivered today and what can be done to improve it.

A. Develop Better Instructional Resources

Improving any educational area requires developing effective instructional resources, ranging from good textbooks to powerful interactive materials, demonstrations, and software tools. Computer vision is currently under significant growth and development. New material is produced every day, while previous material becomes outdated quite fast. The most important educational resource for both educators and students is a good textbook. Maxwell [6], [8] has reviewed some of the most popular computer vision texts available. Most textbooks currently available do not seem to meet the needs of computer science and engineering instructors and students. The main conclusion of Maxwell's review is that textbooks lack an algorithmic and application-based presentation. Moreover, they lack good balance by emphasizing certain areas more than others, making it difficult for instructors to choose a text that covers a variety of topics. In most cases, instruction is based on a set of lecture notes developed by the instructor, while the textbook is meant to be a reference only. The lack of comprehensive textbooks presents serious difficulties in moving the computer vision area forward. Fortunately, in the last few years, several new textbooks have been published, including [96] and [97]. An excellent very comprehensive, textbook by Forsyth and Ponce [33] is also soon to be published. The efforts to improve existing textbooks and write new ones must continue.

Because of the visual nature of computer vision, an important consideration for computer vision educators is to develop interactive materials and demonstrations that will allow students to have active "hands-on" learning experiences. Several educators are currently developing online materials for teaching the traditional courses in image-processing and computer vision (see Section III-B). These efforts need to continue, and the materials developed need to become available to the broader computer vision community for possible adoption and improvements. The role of online computer vision repositories will be invaluable in this respect (see Section VIII-F).

B. Introduce Teaching Innovation

Computer vision is a challenging subject to teach. Introducing innovative teaching approaches can make computer vision courses more effective and rewarding experiences both for the students and for the instructors. Exploiting students' background knowledge, using interactive technology, and embedding research pedagogy are a few examples discussed in this paper (see Section III). Computer vision educators need to consider adopting some of these techniques in their teaching and to continue seeking new ways to produce well-trained and skilled computer vision scientists and engineers.

C. Enhance the Computer Vision Curriculum

Keeping traditional computer vision courses up-to-date and supporting the computer vision curriculum with complementary courses is important for offering comprehensive computer vision education. Several computer vision educators have, in fact, advocated creating entire programs of specialization in computer vision. For example, Jain from the University of California, San Diego, [98] and Kakadiaris from the University of Houston, Houston, TX, [99] have advocated a curriculum on visual computing. The most effective and beneficial way to develop such programs is probably by seeking collaborations with other departments within the same college or across colleges [28], [100]. At the University of Rochester, Rochester, NY, for example, six faculty members from three departments (i.e., Institute of Optics, Electrical and Computer Engineering, and Computer Science) came together to develop a comprehensive undergraduate and first-year graduate curriculum in the broader area of electronic imaging [100]. In the past, each department would offer courses related to some aspect of electronic imaging; however, the benefit to the students was limited. The new program at Rochester contains five new and six enhanced courses, offering comprehensive educational experiences in electronic imaging. Two of these courses are notable: the first is a freshman-level course designed to motivate students to consider electronic imaging as a career, while the second is a special seminar series designed to keep both students and faculty up-to-date on the latest technology in industry and academia.

D. Design Effective Programming/Lab Assignments

Developing effective computer vision programming assignments is not easy. In most cases, the assignments have limited scope and do not expose the students to real-world problems. Hands-on projects, focusing both on software and hardware, would be very useful. It has been argued several times that students need to be trained with a systems approach to acquaint them with cameras, lenses, and lighting [2]. These skills are especially essential for students who plan to follow industry careers. Most students do not know how to take the constraints of the physical world into account or how to adapt a technique to a real application and make it robust. They should be able to analyze a problem, make a decision on how to approach it, try different design procedures, and compare the different outcomes.

Significant emphasis needs to be placed on choosing programming assignments that spark and retain the students' interest and enthusiasm in computer vision. Although it is not difficult to design assignments that deal with real-world problems and have immediate and intuitive results, the main difficulty seems to be with designing meaningful assignments that

are commensurate with the amount of time the students have to complete them.

A recent survey [8] has considered the nature of the assignments used in computer vision courses at 26 different institutions. The results of the survey indicate that designing effective assignments requires sensitivity and careful planning. For example, factors that make effective assignments are

1) choosing the data sets appropriately (e.g., images allowing easy foreground/background separation and feature extraction);
2) making appropriate tools available so as to allow students to finish the assignments successfully;
3) providing satisfactory and worthwhile examples with which students can work.

The survey respondents seem to agree that success in a given task is not necessarily a requirement; however, poor results are not helpful in retaining student interest, especially undergraduate students. Besides improving understanding by reinforcing the concepts and algorithms discussed in class, the goal of the assignments should also be to instill appropriate habits in the students. Exposing them to collaborative work through group-oriented assignments, for example, would be critical for their future careers.

E. Develop Effective Software Tools

Making the appropriate tools available to students is also critical for completing their programming assignments or research successfully and gaining as much as possible from this experience. The lack of a well-accepted software platform for computer vision algorithm development (e.g., like OpenGL in Computer Graphics) is a major issue. Students cannot implement even a simple system if they must implement everything from scratch in one semester. It is very important to provide them with at least some of the basic tools they need to implement their assignments (e.g., edge-detection algorithms).

F. Create Online Computer Vision Repositories

Computer vision is a challenging educational area. When developing a computer vision course, educators must select a text, provide a list of topics, and create a syllabus, lecture notes, homework problems, and programming assignments. The entire process is difficult and time-consuming. The importance of designing effective programming assignments has already been argued. However, their design depends on factors such as the experience of the person developing the assignment othe availability of appropriate equipment to acquire high-quality data sets. As a result, programming assignments related to recent research results might not be an option for many computer vision courses.

A plethora of computer vision instructional material has been developed over the years by computer vision educators all over the world. Although this information could prove to be extremely helpful in developing a high-quality computer vision course, it is not only hard to find but also difficult to use effectively. As a result, the scope, breadth, and quality of a computer vision course might be compromised because most computer vision educators rely on their own educational background. Collecting and organizing this material in a systematic way to make it available to the computer vision educational community would be extremely beneficial.

Maxwell has laid out an ambitious plan for building an online computer vision resource [101].[19] Similar efforts have been reported in other areas, such as in AI [102]. Educators, for example, can share their experiences about what works and what does not. Lecture notes, successful homework problems, computer vision assignments, and data sets can be put on the repository for adoption by others. Solutions can be made available to educators through a password-protected system. Reviews of the material from both the students and the educators can be posted to help improve the resource materials. Creating and, most importantly, maintaining an online resource for computer vision is not an easy task. Such a task will require not only hard work from some very dedicated people, but also support from all who teach computer vision. Supporting the creation of the online resource by contributing these course materials and ideas would be the first step.

G. Organize More Workshops on Computer Vision Education

Conferences and workshops provide a stimulating environment for the exchange of ideas and the development of new ones. The whole issue of improving computer vision education emerged from earlier workshops dedicated to image-related computation [2]–[5]. These workshops generate considerable interest and trigger valuable discussions among many computer vision educators and researchers. The computer vision community needs to continue actively pursuing opportunities to organize workshops and special conference sessions on computer vision education issues in the future.

IX. Conclusion

Computer vision and image computation-related areas have become a pervasive part of modern computing. The applications of computer vision are numerous and range from image databases and human–computer interfaces to medical imaging and robotics. Ten years ago, digital imagery was considered to be an obscure aspect of many computer science and engineering curricula. Today, there is a strong need to train students to become knowledgeable about image-related computation. The goal of this survey has been to review the status of computer vision education today. Hopefully, this review will become a useful source of help and suggestions to educators in computer vision and related areas, triggering further development of the field. This review is by no means comprehensive and is based largely on results that have been reported in the literature and on information found on the World Wide Web. It is certain that more information exists than has been reported in some publication or report. Hopefully, this review will encourage more people interested in this area to step forward and share their ideas and approaches with the rest of the community.

[19]See http://www.palantir.swarthmore.edu/~maxwell/

REFERENCES

[1] K. Bowyer, G. Stockman, and L. Stark, "Themes for improved teaching of image computation," *IEEE Trans. Educ.*, vol. 43, pp. 221–223, May 2000.

[2] "Themes for improved teaching of image-related computation," in IEEE Computer Vision Pattern Recognition Conf. (CVPR'96), San Francisco, CA, 1996.

[3] "IEEE Workshop on Undergrad Education and Image Computation," in conjunction with IEEE Computer Vision Pattern Recognition Conf. (CVPR'97), Puerto Rico, Available online: http://marathon.csee.usf.edu./teaching_resources.html.

[4] "IEEE Workshop on Undergrad Education and Computer Vision," in conjunction with IEEE Computer Vision Pattern Recognition Conf. (CVPR'00), Hilton-Head Island, SC, Available online: http://marathon.csee.usf.edu./teaching_resources.html.

[5] "IEEE Workshop on Combined Research-Curriculum Development in Computer Vision," in conjunction with IEEE Computer Vision Pattern Recognition Conf. (CVPR'01), Hawaii, Available online: http://cs.unr.edu/CRCDWorkshop01.

[6] B. Maxwell, "Teaching computer vision to computer scientists: Issues and a comparative textbook review," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 12, no. 8, pp. 1035–1051, 1998.

[7] B. Draper and R. Beveridge, "Teaching image computation: From computer graphics to computer vision," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 15, no. 5, pp. 823–831, 2001.

[8] B. Maxwell, "A survey of computer vision education and text resources," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 15, no. 5, pp. 757–773, 2001.

[9] A. Sanchez, J. Velez, A. Moreno, and J. Esteban, "Introducing algorithm design techniques in undergraduate digital image processing courses," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 15, no. 5, pp. 789–803, 2001.

[10] P. Wankat and F. Oreovicz, "A different way of teaching," in *Proc. ASEE Prism*, Jan. 1994, pp. 15–19.

[11] D. Coppula, "Integrating teaching and research," in *Proc. ASEE Prism*, Dec. 1997.

[12] (2001) Computing Curricula 2001 – Final Draft. Computer Society of the Institute of Electrical Engineers (IEEE-CS) and the Association for Computing Machinery (ACM). [Online]. Available: http://www.computer.org/education/cc2001/final/index.htm

[13] L. Grewe, "Effective computer vision instruction through experimental learning experiences," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 15, no. 5, pp. 805–821, 2001.

[14] N. Glasgow, *A Guide to Student-Centered, Problem-Based Learning*. Thousand Oaks, CA: Corwin Press, 1997.

[15] P. Ram, "Problem-based learning in undergraduate education," *J. Chem. Edu.*, vol. 76, no. 8, pp. 1122–1126, 1999.

[16] T. Pridmore and W. Hales, "Understanding images: An approach to the university teaching of computer vision," *Engineering Science Education J.*, vol. 4, no. 4, pp. 161–166, 1995.

[17] K. Novins and B. Mccane, "Incorporating primary source material into the undergraduate computer vision curriculum," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 15, no. 5, pp. 775–787, 2001.

[18] M. Donelan and J. Wallace, "Peer assisted learning: A truly co-operative initiative," in *Students Supporting Students*, J. Dolan and A. Castely, Eds. Kogan Page, London, U.K.: Staff and Educational Development Assn., 1998, pp. 11–22.

[19] R. Ploetzner, P. Dillenbourg, M. Praier, and D. Traum, "Learning by explaining to oneself and to others," in *Collaborative Learning: Cognitive and Computational Approaches*, P. Dillenbrourg, Ed. Oxford, U.K.: Elsevier, 1999, pp. 103–121.

[20] P. Mercurio, "Khoros," *Pixel*, pp. 28–33, 1992.

[21] J. Rasure, R. Jordan, and R. Lotufo, "Teaching image processing with Khoros," in *Proc. IEEE Int. Conf. Image Processing*, vol. I, 1994, pp. 506–510.

[22] R. Lotufo and R. Jordan, "Hands-on digital image processing," in *Proc. IEEE Conf. Frontiers Education*, 1996, pp. 1199–1202.

[23] Khoral Home Page [Online]. Available: http://www.khoral.com

[24] S. Moscariello, R. Kasturi, and O. Camps, "Image processing and computer vision instruction using Java," in IEEE Workshop on Undergrad Education and Image Computation, Puerto Rico, 1997.

[25] M. Powell and D. Goldgof, "Software toolkit for teaching image processing," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 15, no. 5, pp. 833–44, 2001.

[26] R. Fisher and K. Koryllos, "Interactive texbooks; Embedding image processing operator demonstrations in text," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 12, no. 8, pp. 1095–1123, 1998.

[27] D. Skocaj, A. Jaklic, A. Leonardis, and F. Solina, "Sharing Computer Vision Algorithms Over World Wide Web," Univ. of Ljubljana, Computer Vision Laboratory, Ljubljana, Slovenia, Tech. Rep., 1996.

[28] M. Sonka, E. Dove, and S. Collins, "Image systems engineering education in an electronic classroom," *IEEE Trans. Educ.*, vol. 41, pp. 263–272, Nov. 1998.

[29] I. Essa and G. Brostow, "A course on digital video special effects," in IEEE Workshop on Undergrad Education and Image Computation, Hilton-Head Island, SC, 2000.

[30] J. Kender, "Visual interfaces to computers: A systems-oriented first course in robust control via imagery," in IEEE Workshop on Undergrad Education and Image Computation, Hilton-Head Island, SC, 2000.

[31] E. Bender, *Mathematical Methods in Artificial Intelligence*. Piscataway, NJ: IEEE Computer Society Press, 1996.

[32] T. Moon and W. Stirling, *Mathematical Methods and Algorithms for Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 2000.

[33] D. Forsyth and J. Ponce. Computer Vision – A Modern Approach. [Online]. Available: http://www.cs.berkeley.edu/~daf/book.html

[34] M. Gini, "Learning computer science through robotics," in ASEE Annu. Conf., Washington, DC, 1996, Session 1626.

[35] W. Clement, "An instructional robotics and machine vision laboratory," *IEEE Trans. Educ.*, vol. 37, pp. 87–90, Feb. 1994.

[36] Z. Liang, "Teaching robot vision in manufacturing technology," in ASEE Annu. Conf., Washington, DC, 1996, Session 1463.

[37] P. Allen, T. Jones, J. Crosby, and P. McCoog, "The virtual vision lab: A simulated/real environment for interactive education in robot vision," in ASEE Annu. Conf., Washington, DC, 1996.

[38] E. Krotkov, "Robotics laboratory exercises," *IEEE Trans. Educ.*, vol. 39, pp. 94–97, Feb. 1996.

[39] E. Wang, "Teaching freshmen design, creativity and programming with Legos and LabVIEW," in *Proc. IEEE Frontiers Education*, vol. 3, 2001, pp. F3G-11–15.

[40] R. Murphy, "Competing for robotics education," *IEEE Robot. Automat. Mag.*, pp. 44–55, June 2001.

[41] ——, "Robots and education," *IEEE Intell. Syst.*, pp. 14–15, Nov./Dec. 2000.

[42] I. Horswill, "A laboratory course in behavior-based robotics," *IEEE Intell. Syst.*, pp. 16–21, Nov./Dec. 2000.

[43] B. Maxwell and L. Meeden, "Integrating robotics research with undergraduate education," *IEEE Intell. Syst.*, pp. 22–27, Nov./Dec. 2000.

[44] K. Sutherland, "Undergraduate robotics on a shoestring," *IEEE Intell. Syst.*, pp. 28–31, Nov./Dec. 2000.

[45] M. Rosenblatt and H. Choset, "Designing and implementing hands-on robotics labs," *IEEE Intell. Syst.*, pp. 32–39, Nov./Dec. 2000.

[46] G. Holmes and T. Smith, "Adding some spice to CS1 curricula," in *proc. SIGCSE'97*, 1997, pp. 204–208.

[47] R. Jimenez-Peris, Sami-Khuri, and M. Patino-Martinez, "Adding breadth to CS1 and CS2 courses through visual and interactive programming projects," in *Proc. SIGCSE'99*, 1999, pp. 252–256.

[48] H. Fell and V. Proulx, "Exploring martian planetary images; C++ exercises for CS1," in *Proc. SIGCSE'97*, 1997, pp. 30–34.

[49] O. Astrachan and S. Rodger, "Animation, visualization, and interaction in CS1 assignments," in *SIGCSE'98*, 1998, pp. 317–321.

[50] R. Montgomery, "Image analysis: A group assignment in programming with breadth," in *Proc. ASEE/IEEE Conf. Frontiers Education*, vol. 2, Nov. 1995, pp. 4d3.12–4d3.14.

[51] G. Stockman and R. Enbody, "Teaching advanced students C++ with computer vision," in IEEE Workshop on Combined Research-Curriculum Development in Computer Vision, Kauai, Hawaii, 2001.

[52] J. Marks, W. Freeman, and H. Leitner, "Teaching applied computing without programming: A case-based introductory course for general education," in *Proc. 32nd SIGCSE Tech. Symp. Computer Science Education*, vol. 33, 2001, pp. 80–84.

[53] D. Egbert, G. Bebis, M. McIntosh, N. LaTourrette, and A. Mitra, "Computer vision research as a teaching tool in CS1," in ASEE/IEEE Conf. Frontiers Education, Boston, MA, 2002, submitted for publication.

[54] G. Allan and J. Zylinski, "The teaching of computer programming and digital image processing in radiography," *Int. J. Medical Informatics*, vol. 50, pp. 139–143, 1998.

[55] S. Sarkar and D. Goldgof, "Integrating image computation in undergraduate level data-structure education," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 12, no. 8, pp. 1071–1080, 1998.

[56] S. Sarkar, "Evaluation of effectiveness of incorporation of computer vision into undergraduate data structures," in IEEE Workshop on Combined Research-Curriculum Development in Computer Vision, Kauai, Hawaii, 2001.

[57] E. Fink and M. Heath, "Image-processing projects for an algorithms course," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 15, no. 5, pp. 859–868, 2001.
[58] D. Stevenson, "Image related applications for a core algorithms course," *Inte. J. Pattern Recognition Artificial Intelligence*, vol. 15, no. 5, pp. 845–857, 2001.
[59] A. Hoover, "Computer vision in undergraduate education: Modern embedded computing," in IEEE Workshop on Combined-Research Curriculum in Computer Vision, Kauai, Hawaii, 2001.
[60] R. Schultz, "Teaching signals and systems through visualization with image processing," in ASEE Annu. Conf., Milwaukee, WI, 1997.
[61] ——, "A practical introduction to digital signal processing through Microsoft Visual C++ and LabVIEW programming," in ASEE Annual Conference, Seattle, WA, 1998.
[62] F. Cohen and A. Petropulu, "The computer vision component in a DSP laboratory," in IEEE Workshop on Undergraduate Education and Image Computation, held in conjunction with IEEE Computer Vision Pattern Recognition Conf. (CVPR'97), Puerto Rico, 1997.
[63] J. Carr, "Using a digital camera to teach the physics of light," in ASEE/IEEE Conf. Frontiers Education, Reno, NV, 2001.
[64] K. Martini, "Digital imaging in teaching structures: A rigorous visual approach," *J. Professional Issues Engineering Education Practice*, vol. 125, no. 2, pp. 56–64, 1999.
[65] J. Hayes and A. Overton, "Imaging as an educational tool in natural resources engineering," in ASEE/IEEE Conf. Frontiers Education, 1995.
[66] D. Thomas, K. Johnson, and S. Stevenson, "Integrated mathematics, science, and technology: An introduction to scientific visualization," *J. Computers Math. Sci. Technol.*, vol. 15, no. 3, pp. 267–94, 1996.
[67] About NIH Image [Online]. Available: http://rsb.info.nih.gov/nih-image/about.html
[68] R. Greenberg, R. Kolvoord, M. Magisos, R. Strom, and S. Croft, "Image processing for teaching," *J. Sci. Educ. Teaching*, vol. 2, no. 3, pp. 469–80, 1993.
[69] R. Greenberg, M. Magisos, R. Kolvoord, and R. Strom, "Image processing for teaching: A national dissemination program," in *Proc. IEEE Int. Conf. Image Processing*, Nov. 1994, pp. 511–14.
[70] R. Jacqueline and R. Greengerg, "Image processing: A state-of-the-art way to learn science," *Educational Leadership*, vol. 53, no. 2, pp. 34–38, 1995.
[71] R. Greenberg, "The IPT project: Image processing for teaching," *J. Technol. Horizons Educ.*, vol. 24, no. 5, pp. 61–65, 1996.
[72] R. Greenberg, J. Raphael, J. Keller, and S. Tobias, "Teaching high school science using image processing: A case study of implementation of computer technology," *J. Res. Sci. Teaching*, vol. 35, no. 3, pp. 297–327, 1998.
[73] R. Greenberg, "Image processing for teaching: Transforming a scientific research tool into an educational technology," *J. Computers Math. Sci. Teaching*, vol. 17, no. 2/3, pp. 149–60, 1998.
[74] S. Tanimoto, "Image processing in middle-school mathematics," in *Proc. IEEE Int. Conf. Image Processing*, Nov. 1994, pp. 501–505.
[75] ——, "Exploring mathematics with image processing," in *Proc. IFIP WCCE*, 1995, pp. 805–14.
[76] ——, "Connecting middle school mathematics to computer vision and pattern recognition," *Int. J. Pattern Recognition Artificial Intelligence*, vol. 12, no. 8, pp. 1053–70, 1998.
[77] ——, "Pixels, numbers and programs: A freshman introduction to image processing," in IEEE Workshop on Combined Research-Curriculum Development in Computer Vision, Kauai, Hawaii, 2001.
[78] G. Stockman, "A first unit computing with images," in IEEE Workshop on Undergraduate Education Image Computation, Puerto Rico, 1997.
[79] "Shaping the Future: New Expectations for Undergraduate Education in Science, Mathematics, Engineering, and Technology," Directorate for Education and Human Resources, Advisory Committee to the National Science Foundation, 1996.
[80] "National Science Education Standards," National Research Council (NRC), Washington, DC, 1996.
[81] M. Shah and K. Bowyer, "Mentoring undergraduates in computer vision research," in IEEE Workshop on Undergrad Education and Image Computation, Puerto Rico, 1997.
[82] ——, "Mentoring undergraduates in computer vision research," *IEEE Trans. Educ.*, vol. 44, pp. 252–257, Aug. 2001.
[83] K. Novins, "A student centered approach to acquiring background knowledge in computer vision," in IEEE Workshop on Combined Research-Curriculum Development in Computer Vision, Kauai, Hawaii, 2001.
[84] S. Khuri and H. Hsu, "Interactive packages for learning image compression algorithms," in *Proc. Annu. Conf. Innovation Technology Computer Science Education*, 2000, pp. 73–76.
[85] A. Biancardi, V. Cantoni, D. Codega, and M. Pini, "An interactive tool for C.V. tutorials," in *Proc. IEEE Int. Workshop Computer Architectures Machine Perception*, 1997, pp. 170–174.
[86] D. Roman, M. Fisher, and J. Cubillo, "Digital image processing – An object-oriented approach," *IEEE Trans. Educ.*, vol. 41, pp. 331–33, Nov. 1998.
[87] J. Robinson, "A software system for laboratory experiments in image processing," *IEEE Trans. Educ.*, vol. 43, pp. 455–59, Nov. 2000.
[88] F. Depiero, "SIPTOOL: The 'Signal and image processing tool' and engaging learning environment," in *IEEE Conf. Frontiers Education*, 2001, pp. F4C-1–5.
[89] D. Braggins, "A report from Ireland: Software for internet vision class and addressing CMOS," *Advanced Imaging*, vol. 13, no. 10, pp. 44–43, 1998.
[90] S. Eddins and M. Orchard, "Using MATLAB and C in an image processing lab course," *Proc. 1st IEEE Int. Conf. Image Processing*, pp. 515–19, 1994.
[91] M. Jankowski, "New courseware modules and software for digital image processing," in American Society for Engineering Education Annual Conf. Exposition, Albuquerque, NM, 2001, Session 1320.
[92] D. Sohi and S. Devgan, "Application to enhance the teaching and understanding of basic image processing techniques," in *Proc. IEEE Southeastcon Conf.: Preparing for the New Millenium*, 2000, pp. 413–16.
[93] H. Hanna, "Development of computer assisted learning to assist in the teaching of image processing and image coding," in *Proc. IEEE Int. Conf. Multi-Media Engineering Education*, 1994, pp. 387–91.
[94] Data Translation Products page [Online]. Available: http://www.datx.com/products/software/gli.htm
[95] G. Donohoe and P. Valdez, "Teaching digital image processing with Khoros," *IEEE Trans. Educ.*, vol. 39, pp. 137–42, May 1996.
[96] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*. Boston, MA: PWS-Kent, 1999.
[97] G. Stockman and L. Shapiro, *Computer Vision*. Englewood Cliffs, NJ: Prentice-Hall, 2001.
[98] R. Jain, "A sequence of courses in visual computing," in IEEE Workshop on Undergraduate Education and Image Computation, held in conjunction with Computer Vision Pattern Recognition Conf. (CVPR'97), Puerto Rico, 1997.
[99] I. Kakadiaris, "Toward a visual computing curriculum at the university of Houston," in IEEE Workshop on Undergraduate Education and Computer Vision, Held in Conjunction with Computer Vision Pattern Recognition Conf. (CVPR'00), Hilton-Head Island, SC, 2000.
[100] M. Kriss, "Bridging departmental barriers in search for a new electronic imaging curriculum," in ASEE Annu. Conf., Charlotte, NC, 1999, Session 1526.
[101] B. Maxwell, "Building an on-line resource for computer vision educators," in IEEE Workshop on Combined-Research Curriculum in Computer Vision, Kauai, Hawaii, 2001.
[102] I. Russell and B. Manaris, "An AI repository as a course development resource," in ASEE Annu. Conf., Washington, DC, 1996.

George Bebis (S'89–M'98) received the B.S. degree in mathematics and the M.S. degree in computer science from the University of Crete, Crete, Greece, in 1987 and 1991, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Central Florida, Orlando, in 1996.

From 1996 to 1997, he was a Visiting Assistant Professor in the Department of Mathematics and Computer Science at the University of Missouri, St. Louis. From June 1998 to August 1998, he was a summer faculty member in the Center for Applied Scientific Research (CASC) at Lawrence Livermore National Laboratory (LLNL), Livermore, CA. Currently, he is an Associate Professor in the Department of Computer Science at the University of Nevada—Reno (UNR), Reno, NV, and Director of the UNR Computer Vision Laboratory (CVL). His research interests include computer vision, image processing, pattern recognition, artificial neural networks, and genetic algorithms. His research is currently funded by the National Science Foundation, the Office of Naval Research, the National Aeronautics and Space Administration, and the Ford Motor Co. He is on the editorial board of the *International Journal on Artificial Intelligence Tools*.

Dr. Bebis is Vice Chair of the IEEE Northern Nevada section and has served on the program committees of various national and international conferences and has organized and chaired several conference sessions.

Dwight Egbert (S'71–M'74–SM'92) received the Ph.D. degree in electrical engineering from the University of Kansas, Lawrence, in 1976.

He is in his fifteenth year as a full-time Academic Engineer with the University of Nevada, Reno, College of Engineering. Currently, he is the Director of Computer Engineering Undergraduate Programs and Director of the College of Engineering Computer Center. He has authored more than 80 original research papers and reports dealing with computer applications in computer vision, remote sensing, image processing, and neurocomputing.

Mubarak Shah (F'02) received the B.E. degree in electronics from the Daewood College of Engineering and Technology, Karachia, Pakistan, the E.D.E. degree from the Philips International Institute of Technology, Eindhoven, The Netherlands, and the M.S. and Ph.D. degrees in computer engineering from Wayne State University, Detroit, MI, in 1979, 1980, 1982, and 1986, respectively.

He is a leading researcher in computer vision, video computing, and surveillance and monitoring, is a Professor of Computer Science and the Founding Director of the Computer Visions Lab at the University of Central Florida, Orlando, and is one of the active researchers in computer vision, video computing, and surveillance and monitoring. He has supervised several Ph.D., M.S., and B.S. degree candidates to the completion of their degree and is currently directing ten Ph.D. and several B.S. degree students. He has worked with more than 150 undergraduate students from 12 universities all over the country under the Research Experience of Undergraduates (REU) program funded by the National Science Foundation for the last 15 years. These undergraduates have coauthored more than 60 research papers, and approximately half of these participants have gone on to graduate schools, seven have written honors in the major theses, six are now faculty members at various universities, and four have started their own companies. He has published close to 100 articles in leading journals and conferences on such topics as visual motion, tracking, edge and contour detection, shape from shading and stereo, activity and gesture recognition, and multisensor fusion. He is coauthor of one book titled "Motion-Based Recognition" (Norwell, MA: Kluwer, 1997) and is an Editor of the international book series "Video Computing" (Norwell, MA: Kluwer, 1997). He is an Associate Edtior of *Machine Vision and Applications and Pattern Recognition* journals and has been a Guest Editor of a special issue of the *International Journal of Computer Vision on Video Computing*. His research has been funded by the National Science Foundation, the Defense Advanced Research Projects Agency, the U.S. Army, the State of Florida, and several companies, including Boeing, Lockheed-Martin, and the Harris Corp.

Dr. Shah was an IEEE Distinguished Visitor speaker from 1997 to 2000 and is often invited to present seminars, tutorials, and talks all over the world. As a student, he was awarded a five-year Quaid-e-Azam scholarship. He received the Harris Corporation Engineering Achievement Award in 1999; the Teaching Incentive Program Award in 1995; and the IEEE Outstanding Engineering Educator Award in 1997. He was the Program Co-Chair of the IEEE Workshop on Motion and Video Computing, held in Orlando, FL, in December 2002 and has served on program committees and chaired sessions at numerous international conferences and workshops. He is an Associate Editor of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE from 1998 to 2002.