2006-2485: A HANDS-ON, INTERACTIVE UNDERGRADUATE DIGITAL IMAGE PROCESSING COURSE

Agnieszka Miguel, Seattle University

A Hands-On, Interactive Undergraduate Digital Image Processing Course

Abstract

This paper describes an experimental undergraduate digital image processing course created and taught by the author. The course was designed to be an interactive experience. The lecture material, hands-on examples, and in-class computer exercises were blended together to form a unique interactive learning experience. Lectures contained numerous MATLAB-based examples and students were required to experiment with short programs during the presentation. Each class period included a longer computer exercise designed to give students the opportunity to practice the material presented in the lecture. The computer exercises used MATLAB with its Digital Image Processing Toolbox. This paper describes the course in detail and offers practical advice and suggestions for future improvements.

Introduction

The field of image processing has grown tremendously in the last decade. Countless applications of digital image processing (DIP) from personal entertainment to medical and scientific discovery drive the need for graduates with experience in imaging. Multimedia industry, robotics, manufacturing, medicine, and remote sensing are only a few examples of specializations that demand students educated in image processing and computer vision[1].

Nowadays, most image processing courses are taught at the graduate level. However, offering an elective image processing course during the junior or senior years of undergraduate studies has the potential to trigger students' interest in a new and exciting field. It shows them the real world application of the many hours of engineering foundations and fundamentals they had to take during their freshman and sophomore years. Image processing is an excellent choice for their first elective course because results of the algorithms are readily available for visual inspection. Some students may lack the prerequisites to fully understand digital processing of two dimensional signals; however, with some effort from the instructor, the course can be structured to provide the required background. Alternatively, the instructor may choose to evaluate the general knowledge that the students have mastered by the time they are ready to take the image processing class and structure the course around that knowledge. This enables the undergraduate students to sample the world of engineering applications early on and hopefully excites them to pursue this topic in the future. The author has always been an advocate for bringing more real world applications into the early years of electrical engineering education to motivate students and increase the retention. Making image processing accessible and appealing to a wide range of students fits well with this philosophy.

Traditionally, image processing courses have been taught in the format of lectures followed by midterm and final examinations. In such courses, image processing is taught theoretically, with a rigorous mathematical treatment. However, it is also a very practical field; the mathematical formulas can be easily translated into algorithms which can produce visual results in a short period of time. Sage and Unser[6] advocate complementing the pure theory of image processing with practical applications through computer laboratories. They stress that when theory is combined with practical experiments, students develop better understanding and retention of the material. In addition, students are more motivated to study the theory if they can experiment with algorithms[6]. There are two levels of hands-on activities in image processing courses. In the first one, students are asked to apply the algorithms to real world images to see the results. In the second type of activities, students program the algorithms themselves[6] or create a large image processing application by combining several building blocks. In the course described here, the author experimented with both types of hands-on activities. The students were applying the algorithms to real images in the in-class computer exercises and were implementing the algorithms or building complex image processing applications as part of their homework computer assignments and the final course project.

When hands-on experimentation is implemented in image processing courses, it is usually via computer laboratory assignments done after the class meets. However, in the author's opinion this "waiting period" between the time the knowledge is acquired and the time it is applied through hands-on activities in unnecessary and may negatively affect student learning. Students are more likely to understand and retain the theory if it is illustrated with immediate hands-on experiments. In the course described here, students were given the chance to practice the theory at the same time as they were learning it.

The remainder of the paper is organized as follows. The context of the DIP course is first described. Next, the course structure and the material covered are explained. This is followed by the account of the hands-on activities incorporated into the class. The final project and other assignments are then discussed. The paper concludes with a reflection on the parts of the course that were successful. Suggestions about possible modifications to the future course offerings are also given.

Course Context

Seattle University is a comprehensive university serving approximately 6800 undergraduate, graduate, and professional students. The Department of Electrical and Computer Engineering (ECE) offers an exclusively undergraduate program in which students may follow either of two curricular tracks leading to the bachelor's of science in electrical engineering (BSEE) degree. The Electrical Engineering track offers a broad introduction to the entire profession. The specialization in Computer Engineering allows students an opportunity to focus on hardware and software aspects of modern computing. The electrical engineering program presently has an enrollment of approximately 90 students. There are seven faculty members. This allows for a close association between faculty and students and leads to small class sizes. Most elective courses in our department are offered every two years.

The digital image processing course was taught during the 10 weeks of Spring Quarter 2005. This was the first time that the author taught a digital image processing course and the first time that this course was offered at Seattle University. The class met biweekly for 125 minutes. There were 29 students in the class: 3 from the Computer Science and Software Engineering (CSSE) Department (all seniors) and 26 from the Electrical and Computer Engineering (ECE) Department (15 juniors and 11 seniors). The only prerequisites for the class were Fundamentals of Computer Science I and a junior or senior standing in the ECE or CSSE departments. The latter requirement ensured that students have taken freshman and sophomore level math courses as well as Circuits I and II.

Traditionally, our students take most of their elective courses during their senior year. Opening the DIP course to juniors provided our students with more choices in elective offerings during the junior year, introduced them to an important topic that is not covered in any of our other courses, and hopefully stimulated their interest in a new field and built their confidence in their knowledge. Since DIP is a topic of interest to students in both the ECE and the CSSE departments, the course was also open to students from both departments.

Course Structure

There were no exams in this course. Students were given written homework assignments (20% of the course grade), computer projects (25%), and a final project (40%). In addition, students were graded on class participation based on the MATLAB diary of their in-class computer exercises (15%). Students were given one week to work on each homework assignment and two weeks for each computer project. The in-class computer exercises were supposed to be completed by the end of each class period. Students were encouraged to work on the final project throughout the whole quarter. The homework and computer assignments were based on individual work, while the final project was team based (there were 13 teams of two students and one team with three students).

Course Textbook and Topics

The textbook selected for the course was "Digital Image Processing" by Gonzalez and Woods[1]. It is one of the most popular image processing textbooks. However, many students complained that it was too advanced and difficult to read. The author found that the two MATLAB-based digital image processing textbooks[4,5] provided excellent supplementary material, however they were often too superficial in covering the theory of imaging algorithms to be used as primary textbooks.

The topics of the course were as follows:

- Image acquisition methods and representation standards.
- Image processing with MATLAB.
- Human visual perception and color space.
- Image enhancement in the spatial domain (thresholding, contrast stretching, and histogram operations).

- Spatial filtering (smoothing, order-statistics filters, sharpening).
- Dithering.
- Frequency domain image enhancement.
- Image filtering in frequency domain (low pass, high pass, notch filter, and Laplacian).
- Image restoration (types of noise and denoising).
- Morphological image processing (color and gray scale).
- Image segmentation (point and line detection, edge detection, connected components, the Hough transform, region growing, region splitting and merging, watershed and distance transforms).
- Geometric operations (affine transformations and interpolation).
- Color image processing.
- Image compression (lossy vs. lossless, bit plane coding, JPEG).
- Video coding (motion compensation, MPEG family coders).

Hands-On Experience

Most imaging courses at other universities use a traditional lecture format with individual student work limited to the time outside of class. In the DIP course presented in this paper, students had the opportunity to experience hands-on learning every time the class met. The course was designed to be an interactive experience and was taught exclusively in a computer lab. The teaching lab was equipped with 34 Dell Optiplex GX260 Pentium 4 computers for student use, one Dell GX260 instructor's station connected to twin video projectors for classroom presentations, and one HP 9000 LaserJet printer. All computers were connected to the student network, email, and the Internet. They were all equipped with the latest version of the MATLAB programming environment and the MATLAB Digital Image Processing Toolbox. There were enough workstations in the room; students did not have to share the computers. The classroom design supports student-instructor interaction as well as collaboration among students.

MATLAB was selected as the software platform for the course because of several reasons. It is a matrix oriented computing and programming engine and since images can be thought of as matrices, handling images in MATLAB is very straightforward. MATLAB is an interactive programming environment, that is, students can see the results of their commands immediately. The interactive characteristics of MATLAB encourage "learning by discovery"[2]. The simple programming language and the availability of numerous built-in functions make MATLAB the first choice of a programming environment for electrical engineering departments across the world. Most of the built-in functions are written in MATLAB and therefore they can be inspected by students. This way they can learn how a particular algorithm is implemented. In addition to the functions available in the standard distribution of MATLAB, there are numerous user-contributed imaging functions that can be downloaded from the Mathworks web page[7].

Some of the students in the course were not familiar with MATLAB at the start of the quarter. The author had to spend few class periods lecturing about MATLAB. Tutorial materials were distributed and simple exercises were used to gauge if the students are ready to move on to work on images. Note that in the future our department is planning to offer an introductory freshman level MATLAB course which should eliminate the need to spend time at the beginning of each course to teach MATLAB.

The lecture material, hands-on examples, and in-class computer exercises were blended together to form a unique interactive learning experience. All lectures (delivered via PowerPoint) contained numerous MATLAB-based examples and students were required to experiment with short programs during the presentation. Each class period included a longer computer exercise designed to give the students opportunity to practice the material they just learned.

The author experimented with two methods of incorporating interactive exercises into the class. In the first method, the exercises were interleaved with the lecture. The lecture was interrupted several times to allow time for exercises that addressed the topic just covered. Once the students completed the exercise, the lecture was continued until the next exercise and so on. The author believes that this method is superior in terms of student learning because it breaks the monotony of the lecture with hands-on activities. However, the drawback of this method is that in larger classes, it is difficult to give feedback to everybody in the short period of time allocated for such exercises. In addition, some students need more time than others to complete the exercise. The author noticed that some students were done very soon and were waiting aimlessly for the lecture to resume while others were straggling to finish the exercise on time. This method may have a negative impact on the slower students as they may lose confidence in their abilities.

The second method is based on a longer lecture with MATLAB examples; however, the students are not prompted to do any exercises until the lecture is over. This ensures that students who are slower are given a long period of time during which they can work on the exercise. The total time spent on the exercises is the same. However, in this case, students can relax and concentrate on understanding the exercise instead of stressing out that they are not able to complete it on time.

The exercises were of three types. The first type was a guided exploration of a particular topic aimed at providing in-depth understanding of the concepts learned during the course. For example, in the edge detection module, students were asked to compare different edge detection techniques, comment on the parameter selection, and on the choice of the algorithm for a particular image. This was the simplest type of an exercise, designed to strengthen student's understanding of the lecture topic and improve their confidence in using MATLAB to design DIP applications.

In the second type of exercises students were directed to answer a question, often in a competitive way. For example, in the image restoration module, students were given copies of an image corrupted with different types of noise. First, they were asked to find the type of noise for each image. Then, based on their answer to the first question, they had to remove as much noise as possible. This task was posed as a competition where the results were judged by computing the mean squared error between the recovered and the original images.

Finally, in the third type of an exercise, students were interacting with each other. For example, in the segmentation module, each student was asked to hide a couple of high amplitude pixels in an image and pass the altered image to his/her colleague who was supposed to find the location of the high amplitude pixel using point detection techniques.

Mixing the three types of activities made the class more interesting and allowed students to practice the material learned in the lecture. Such in-class exercises give an immediate feedback to the students on how well they understood the concepts presented in class. The material is still fresh in their mind. Using it immediately in a practical setting improves its assimilation and retention. The instructor was available during the time the students worked on the exercises to give feedback, address important issues, and correct any misunderstandings.

There were twenty in-class computer exercises (at least one for each module). The topics of the exercises are listed below:

1. Introduction to MATLAB (five exercises):
    - Setting up the MATLAB environment. Using the diary command.
    - Reading images into the MATLAB workspace.
    - Displaying images in MATLAB.
    - Writing images to the hard drive.
    - Array indexing in MATLAB.

2. M-functions programming:
    - The exercise asked students to write a program that generates a test pattern image. The size of the image and the width of the stripes were the input variables of the program.

3. Human visual perception and color space:
    - The exercise asked students to separate the three color planes in an RGB image and display them as gray scale images. Then, the students combined two planes at a time and commented on the results. Finally, they experimented with converting the RGB image into different color spaces (HSV, YCbCr).

4. Image enhancement in the spatial domain:
    - Students experimented with generating a negative version of the input image. Then, they were asked to apply log transformation, contrast stretching, and histogram equalization to an image. They commented on the role of different parameters in these algorithms.

5. Spatial filtering:
    - The exercise asked students to apply an averaging filter of an increasing size to an image and comment about the results. The

students also added salt and pepper noise to the image and used median filtering to remove it. They also experimented with the Laplacian filter, unsharp masking, high-boost filters, and Sobel and Roberts operators.

6. Dithering:
    - This was a very interesting exercise in which students thresholded a gray scale image to produce a black and white version of the image. Then, they added noise to the original image and thresholded it. When the two results were compared, it was obvious that the second method produced a more natural looking black and white image.

7. Frequency domain image enhancement:
    - Students experimented with computing the Discrete Fourier Transform of simple images containing an edge or a box in order to gain understanding of image representation in the 2D frequency domain.

8. Filtering in the frequency domain:
    - The exercise asked students to experiment with low pass filters of a different size. Then, they added a periodic noise to the image and attempted to remove it using frequency filtering methods.

9. Image restoration:
    - Students were given several copies of the same image corrupted with different types of noise. They were asked to recognize the noise type based on the shape of its frequency spectrum. Then, they were using different filtering methods to remove as much of the noise as possible (measured by the MSE).

10. Mathematical morphology:
    - The exercise asked students to increase the size of the structuring element until some structures in the input image were removed while other (thicker) structures were still preserved. Then, they were asked to use the hit-and-miss transform to find the location of certain features in another image.

11. Segmentation (point, line, and edge detection):
    - Students were asked to add a bright pixel to an image of their choice and pass it to another student who had to find out the location of the spot using point detection techniques. Then, students were challenged to isolate lines in certain images.

12. Segmentation (connected components, Hough transform, and adaptive thresholding):

    - Students were asked to use the connected components algorithm to label elements in a picture. They also experimented with line detection in an image using the Hough transform.

13. Segmentation (region growing, region splitting and merging, and watershed segmentation):
    - The exercise asked students to use the above algorithms with appropriately modified parameters to correctly segment features in medical and scientific images.

14. Geometric operations:
    - Students were asked to comment on the quality of a thumbnail-size version of an image produced using different interpolation methods. They also experimented with rotation and image enlarging.

15. Image compression:
    - Students were asked to compress a binary image using run length encoding (RLE) and compute the resulting compression ratio. Then, they applied the discrete cosine transform to an 8x8 block from an image, quantized it with different quantization factors, and encoded the result with RLE.

The in-class computer exercises can be thought of as classroom assessment techniques. The instructor can observe how well the students are doing and modify her/his teaching methods to improve student learning. Students were graded for class participation based on the completion of the in-class exercises. They were required to turn in a diary of their everyday in-class work. (The command *diary* in MATLAB was used for this purpose.) However, this was more of an "all or nothing" score as the diary code is quite difficult to grade. Unfortunately, this check point was necessary to make sure that all students were involved in the exercises.

Final Project

The course culminated in a team-based project. The goal of the final project was to demonstrate the ability to apply the knowledge and the techniques learned during the course to solve a digital image processing problem. All projects involved writing a MATLAB program (using the Image Processing Toolbox). The final project required selecting a topic from a list provided during the first week of the quarter or proposing a new topic, writing a project proposal (4% of the course grade), demonstrating the program (12%), writing a project report (12%), and preparing and presenting a poster (12%). The project counted for 40% of the class grade. The project topics were as follows:

- Image fusion for multifocus images (digitally superimpose two or more color pictures of the same object or a scene at different focus settings).
- Photoquantigraphic imaging (extend the dynamic range of an image by combining differently exposed images).

- Visual disabilities demo (write a program which modifies an input picture in such way that people with normal vision can see how a person with a particular disability would perceive such an image).
- Image mosaic (combine a subset of small images into a new full-size image that when viewed from a distance resembles an input picture).
- Information hiding in images (implement and compare the results of a few methods that embed information in an input image).
- Artificial painting toolbox (research and implement algorithms that process a photograph to make it look like a painting).
- Special effects toolbox (implement several special effects transforms for color images: radial pixilation, ripple effects, fisheye, twirls, solarization, fuzzy effect, random tile effect, etc).
- Stained glass effect (design and implement an algorithm that generates a stained glass effect for an input image).
- Counting car colors (create a database of digital images of cars driving on a highway and write a program that automatically finds cars in the images and creates a histogram of all car colors that are present in the image set).
- License plate segmentation (create a database of digital images of cars on a parking lot and write a program that automatically finds the license plate in each image).
- Image panorama (create a database of images that cover a long stretch of a landscape or city view and write a program that generates a panoramic image from this set of images).
- Image processing for ROBOCOP (segmentation and identification of color-coded robots).
- Automatic color balancing (correct a series of underwater images by compensating for the over saturation of the blue channel).
- Enhancing visibility of digital game objects (write a program that helps visually impaired players see the special hidden objects that exist in the background).

Students were encouraged to work on their final projects throughout the whole quarter. They were given the list of proposed projects during the first week of the quarter. Students were allowed to form their own teams; the team selection had to be accomplished by the end of the first week of classes. Project selection was due by the end of the second week of the quarter. Note that by that time, students do not have enough background to fully understand what the different projects entail. However, with the use of the textbook, the Internet, and advice from the instructor, they were all able to select their project. The project proposal was due by the end of the third week of classes. This ensured that students were involved and thinking about their project from the beginning of the class. Asking the students to explain what their project is about and propose what they are going to accomplish forces them to research the particular subject early. As new material was introduced throughout the quarter, students were able to make connections with their early research and assimilate the newly learned knowledge in the context of their own project.

The final deliverables of the project included program demonstration, poster presentation, and final report. Program demonstration was often done outside of the class time, at a time convenient to the team and the instructor. The purpose of this demonstration was to prove that the program runs correctly. It was also an opportunity for the author to get a sense of the level of each student's contribution to the program and her/his understanding of the theory and implementation details. In addition, such demonstration forces students to consider user interface and a proper choice of test cases. Poster presentation was done during the final exam period. Each team was responsible for the preparation of a poster that illustrated their project. The poster had to include the background about the project topic, description of the algorithm, discussion of the results, and conclusions. The teams were also asked to prepare a 3-minute summary of their project that would be delivered using the poster. The poster session was a very successful event; students were very interested in learning what the other teams did for their project and many questions were asked. Microsoft Publisher was used to design the posters. The department covered the printing cost of the posters. The posters are currently on display in one of our labs.

The final report was an opportunity for the students to document what they have learned about their particular topic, describe the details of the algorithm(s) that they have implemented, comment on their performance and results, and give conclusions and recommendations for future work. Students were asked to convey their findings in a clear and concise manner and were given guidelines about the technical report format to follow.

The final project was a very successful aspect of the class. The author tried to expose the students to real-world digital image processing applications by asking them to apply theory learned in class to solve practical imaging problems. The students were also given the chance to experience problem solving in a collaborative environment. Such early exposure to group-oriented project assignments is likely to help them in their professional life.

Students commented that the time spent working on their projects felt more constructive and worthwhile than the time spent studying for exams in other courses. In the author's opinion, projects let students truly use and enjoy the knowledge learned in class and put their skills and understanding to a test where creativity and hard work is appreciated and rewarded. Projects also help to put what they learn in perspective and they install a sense of ownership about the student's knowledge. Students enjoy the process of creating something meaningful (software application) because it gives them a sense of accomplishment, especially when compared to a more passive final exam.

Other Assignments

Students were required to complete traditional homework assignments which involved both qualitative and quantitative questions that tested their understanding of image processing algorithms. These were pen-and-paper types of exercises and therefore they had to be limited to very small image matrices.

In addition, students were given computer assignments in which they were asked to solve a particular image processing task for a given data set. The computer assignments were more involved than the in-class computer exercises and required a significant time investment on part of the students. These computer assignments were designed to give students the opportunity to prove themselves at solving an imaging problem and as a result gain confidence in their knowledge and encourage them to start working on the final project.

Evaluation and Assessment

This is the first time that this course has been offered at Seattle University and the first time that the author has taught it. Therefore, it is not possible to compare the learning outcomes of this course to its previous offerings in order to illustrate how the course structure and hands-on activities improved student learning.

Each course taught in the Department of Electrical and Computer Engineering has a number of outcomes associated with it. As part of the course syllabus, students are given the course outcomes table at the beginning of the quarter. At the end of the quarter, students are asked to fill out a survey that indicates their self-perceived level of accomplishment on each outcome using a holistic scale rated 1-5:

1. Topic was not covered.
2. Topic was introduced. I know what it is. I know the basic definitions and notation.
3. Topic was covered. I am familiar enough with it that I could read about it with understanding and apply the knowledge gained to the solution of a problem.
4. Topic was well covered. I have a basic working knowledge of how to apply the principles to a variety of problems.
5. Topic was thoroughly covered. I can use it with some confidence and interpret the results.

Students are also able to make written comments. The course instructor independently assesses student accomplishment using the same holistic scale and justifies the assessment with hard data (test problems scores, project grades, etc). It should be noted that students are not expected to achieve a top level of accomplishment for every course outcome. A few topics are merely introduced. Considerable time is devoted to exploring other topics, and a correspondingly higher level of accomplishment is expected. The results of the outcomes survey conducted in the DIP class are summarized in Table 1. The shaded cells in the course outcomes tables indicate the faculty's agreed upon expected level of student accomplishment.

As the table shows, many of the course outcomes were achieved at a satisfactory level. In the case of outcomes 3, 4, and 7, the score for the student self-reported achievement was significantly lower than what was expected of them. The author believes that the lower achievement on those topics resulted from the lectures not stressing enough the theory behind the corresponding algorithms. Instead, students were expected to use MATLAB functions which resulted in their lack of confidence with regard to their ability to implement these algorithms.

The students were generally very satisfied with the course. They found the in-class hands-on computer exercises extremely useful. They also liked the final project even if they perceived it as more time consuming than studying for a final exam in other classes. The DIP class proved to be effective and rewarding class both to the students and to the instructor.

Table 1. Course outcomes summary for the digital image processing course.

Course Objectives A student will:	Expected Level of Accomplis hment	Student Self-Reported Average	Instructor's Rating of Students
1. Explain binary, gray scale, and color image representation standards.	**5**	**4.5**	**5**
2. Understand human visual perception, and the concept of color space.	**4**	**3.7**	**4**
3. Compute image characteristics such as the histogram.	**5**	**3.8**	**5**
4. Implement histogram equalization.	**5**	**3.8**	**4**
5. Understand dithering.	**3**	**3.5**	**4**
6. Perform smoothing and sharpening of an image.	**5**	**4.1**	**5**
7. Implement edge detection.	**5**	**4.0**	**4**
8. Implement image segmentation.	**4**	**3.9**	**4**
9. Perform morphological operations on an image.	**4**	**4.2**	**4**
10. Understand image and video compression.	**4**	**3.4**	**3**
11. Use a computer tool to design and build image processing application.	**4**	**3.9**	**4**
12. Experience problem solving in a collaborative environment.	**4**	**3.6**	**4**

Lessons Learned

There are several aspects of the course that could be improved. First of all, the in-class exercises can be modified to be more interesting and engaging. Students enjoy friendly competition and an interaction with each other. With that in mind, some of the exercises could be framed as contests and rewritten to include more student-to-student interaction.

Another component of the class that could be improved is the relationship between the length of the lecture and the length of the in-class exercise. As explained before, long lectures followed by longer in-class computer exercises worked better than short sections of lecture interleaved with short exercises. The author still desires to break lectures into smaller sections because she believes that this helps student learning. The author would like to investigate ways to combine a series of short lectures with short exercises without hurting students who need more time to finish the exercises.

In addition, the number of homework computer assignments could be increased. These assignments are important because they teach students problem solving skills which they will need for the final project. Some of the in-class computer exercises could be redesigned and used as a lead into a longer homework computer assignment. Students who finish their in-class exercise earlier could have an early start (and access to the instructor's comments) on their computer assignment.

Finally, another way to break the long lectures would be to have more in-class exercises where students are required to prove their understanding of the algorithm without the use of MATLAB. Using a high-level programming language with imaging libraries that are readily available may hide the details of how the algorithms work behind a user-friendly function call. It was the author's impression that sometimes students were not willing to learn how the technique works, instead, they were relying on MATLAB to do the work for them. Relying too much on MATLAB is a potential drawback of such courses and one way to combat it is via more paper-and-pen exercises. The author plans to scale back on the number of techniques introduced for each topic. Instead a selected main technique will be covered in detail including the underlying theory with a rigorous mathematical treatment. This will shift the course from covering a broad number of techniques at a rather superficial level to a more analytical framework with in depth understanding of selected algorithms. Students can use this in depth knowledge of a particular algorithm as a starting point to understand other algorithms in the same area of image processing. In fact, the other algorithms could be included as topics of the final project or computer assignments. Such teaching methodology of moving back and forth from a particular algorithm to more general aspects of the topic is considered to be very effective for student learning[2].

It should be noted that designing such course requires a considerable amount of time from the instructor. Significant time has to be spent creating PowerPoint slides and writing and verifying the in-class exercises, and homework computer assignments. The author estimates that it took about eight hours per one two-hour lecture to prepare this course. The level of effort required increased during the last weeks of the quarter when students were nearing the completion of their project and ask a lot of questions about their programs.

Additionally, the level of MATLAB knowledge among students in the class was very uneven. The author had to take extra care to provide MATLAB help to the students who needed it and keep students with superior programming skills interested enough to explore additional topics after they completed the basic exercise. The course started with a review of MATLAB; however this was not enough time for some students to get comfortable with programming in this environment. In the future, the author plans to take advantage of a new, introductory MATLAB class that is going to be offered at Seattle University and require this class as a prerequisite for the digital image processing course.

Conclusion

This paper describes an experiential digital image processing class offered for the first time at Seattle University. The purpose of this elective course was to provide students with practical skills and analytical background for building digital image/multimedia applications. The objectives of the course were for the students to gain the knowledge and practical experience in digital image processing. This was achieved through an extensive use of hands-on in-class computer exercises and a final team-based project which demonstrated students' ability to apply the knowledge and the techniques learned during this course to solve a particular research or industry problem. The paper comments on the successful aspects of the course and gives recommendations about future improvements.

References

1. George Bebis, Dwight Egbert, and Mubarak Shah, "Review of Computer Vision Education," *IEEE Transactions on Education*, vol. 46, no. 1, pp. 2-21, February 2003.
2. Steven L. Eddins and Michael T. Orchard, "Using MATLAB and C in an image processing lab course," *Proceedings of the IEEE International Conference on Image Processing*, vo.l 1, pp. 515-519, 1994.
3. Rafael C. Gonzalez and Richard E. Woods, "Digital Image Processing", 2nd edition, Prentice Hall, Upper Saddle River, New Jersey, 2002.
4. Rafael C. Gonzalez, Richard E. Woods, and Steven L. Eddins, "Digital Image Processing Using MATLAB", Prentice Hall, Upper Saddle River, New Jersey, 2004.
5. Alasdair McAndrew, "Introduction to Digital Image Processing with MATLAB," Course Technology, a division of Thomson Learning, 2004.
6. Daniel Sage and Michael Unser, "Teaching Image-Processing Programming in Java," *IEEE Signal Processing Magazine*, vol. 20, no. 6, pp. 43-52, November 2003.
7. MATLAB Central, http://www.mathworks.com/matlabcentral/.