Impact of Piggybacked MATLAB in C-programming Course

Dr. Maddumage Karunaratne, University of Pittsburgh, Johnstown

Dr. Maddumage Karunaratne is an Associate Professor and the Head of the Electrical Engineering department at the University of Pittsburgh at Johnstown, PA. The department offers two undergraduate degrees in Electrical Engineering and Computer Engineering. Dr. Karunaratne earned a Bachelor of Science degree from the University of Moratuwa, a Master of Science from the University of Mississippi, and a Ph.D. from the University of Arizona.

Before joining academia, he gained fourteen years of extensive industry experience in Silicon Valley working in the semiconductor industry performing software development, application engineering, design, testing and verification of digital integrated circuits. He has taught electrical and general engineering classes at Pitt-Johnstown since 2004.

His research and teaching interests include Semiconductor circuit Testing and Verification, Low Power Design Analysis, Digital and Embedded Systems, Electromagnetic Wave Scattering, and IC Design Automation Software development. He has authored or coauthored 26 publications and he holds one US patent and another under review.

He can be reached at maddu@pitt.edu 225 Engineering and Science Building University of Pittsburgh at Johnstown Johnstown, PA 15904

Impact of Piggybacked MATLAB in C-programming Course

Abstract:

Today, an increasing number of scientists and engineers are spending more and more of their work hours in front of the computer. Electronic and semiconductor industry are making capable and inexpensive portable consumer devices as evident from smart phones and tablets that are coming out to the market at an accelerated phase. Manufactures have made attempts to launch hobby industries around inexpensive electronics, particularly processor boards, with more capabilities and easy to program systems such as Raspberry Pi. For individual developer or capable consumer those devices offer customization to a level that was never seen before. However, such customizations require development of computer programs to control the devices and data streams. When electrical and computer engineers are trained, it is becoming more imperative that nearly all acquire some level of computer programming skills to effectively function as engineers in their careers. The nature of work performed in industry changes as they progress in careers. Lack of programming ability and experience may challenge their opportunities for technical and even managerial advancements. For example, a senior engineer without programming experience would not become a project manager if that project requires a significant amount of software to be developed in house.

At this university, electrical engineering technology (EET) and computer engineering technology (CET) majors always take one semester course on computer programming so they can be effective in using embedded controllers and other programmable devices, later in their curricula or in industry after graduation. It is a C based programming course with few projects appropriate for second year engineering technology students. CET majors further study Java, Ada, and C flavored languages in their curricula. EET majors do not have opportunity to learn other languages although some of them would program – using proprietary languages- Programmable Logic Controllers (PLCs) after graduation in industrial settings.

Several of follow on courses taken by both majors require them to use MATLAB as a problem solving tool in advanced circuit theory and control systems theory courses. Several years ago at this university, students had been learning basic MATLAB on their own, and then learned advance features such as control and signal processing toolboxes with help from instructors in follow-on courses. Instructors in the upper level classes could only make limited efforts to help students learn MATLAB. Their efforts are geared toward actual subject contents which are heavy in abstract concepts and mathematics. The author introduced MATLAB in C programming course in the fall of 2012 with the intent of reducing the future burden of learning its basics on their own. That experience has already been published[3] by the author.

This paper discusses the continuing experience in having MATLAB as an additional programming tool to sophomore level students who are learning programming in C language as their main objective. They still learn advanced concepts and toolboxes in higher level courses. An additional benefit expected and clearly seen over the years is that MATLAB reinforces concepts taught in C such as loops, indexing, conditionals, input/outputs, storage and file management, data and program

structures, etc. Also, learning it at this stage to create animation programs provides incentives and variety to further practice algorithm development and problem solving skills. Practice of such skills is essential to become competent programmers.

In addition to the findings already discussed in the previous paper, this paper presents survey results from students in several follow-on courses after they have taken this basic C course with piggybacked MATLAB content. The conclusion provides feedback from students as they progress through follow on courses clarifying the cost-benefit of the enhancements done to the basic C programming course a few years back.

Part I: introduction:

Since the transistor was invented many decades ago, electronic industry continues to make vast strides in providing functionally rich inexpensive devices for consumers and all industries. In recent years that trend has even accelerated due to very inexpensive integrated circuit manufacturing technology which can now implement entire electronic systems in a single integrated package containing several billion transistors. Silicon vendors, having matured scalable technologies, now offer processors with very high performance computing power and data bandwidth at low prices launching niche segments in unmanned aerial vehicles, surveillance and monitoring systems, home automation, GPS tracking, etc. Such inexpensive devices have been proliferated in to almost every consumer device, equipment, or instrument that can provide some level of electric power.

Engineers and scientists heavily depend on computing devices with specialized computer programs in their work. While majority of them may not be willing to create their own software or become software developers, they would still need to understand the trends, adapt, and adopt the technologies to be successful in their careers. Teaching embedded systems has been around for a long time for the purpose of giving the skills to develop such control and communication systems in both software and hardware. However, the value of acquiring such skills has even been increased now that almost every controllable device, from expensive automobiles to inexpensive household LED light bulbs, contains embedded devices for customization and interconnectivity. The recent buzz word for such devices is the Internet Of Things (IOT). Some of them may even provide interfacing capability to smart phones via custom developed small software modules known as Apps. Such customizations provide ample opportunities to electrical and computer engineers confident in computer programming who can then take on large projects as developers or lead personnel. This adds further reasons to make electrical and computer students confident in computer programming.

The programming language and the platform of choice for embedded processors is mostly C or C++ due to its efficiency in memory allocation, run time and ability to directly manipulate data in hardware components although some languages such as Python has gained some popularity for embedded processors. Therefore, teaching C/C++ to electrical and computer engineering students is considered essential in any undergraduate curricula. Once the solution is finalized in an algorithm, the software program may be developed for a specific platform such as MS Windows, Androids, etc. Such developments can be done in most software packages (CGI, MATLAB, JAVA, MS Studio, MS NET, etc.). For hardware (electrical and computer) engineers, their main programming domain still remains as C/C++, which is the main reason C language is being taught and required by electrical and computer engineering programs at this campus. Further, it is easy to learn other programming languages after one has learned programming in C or C++.

Prior to introduction of MATLAB to this course, students learned basic MATLAB on their own, and then learn advance features in control and signal processing toolboxes with some help from instructors in follow-on courses. Instructors in the upper level classes can only make limited efforts to help students learn basics in MATLAB if it were to be taught from scratch. Their efforts are geared toward actual subject contents which are heavy in abstract concepts and mathematics. However, MATLAB toolboxes, and advanced and general features are required in follow-up courses.

This paper discusses the continuing experience of introducing MATLAB as an additional programming tool to sophomore level students who are learning programming in C language as their main objective. The author introduced MATLAB in C programming course in the fall of 2012 with the intent of reducing their future burden of learning MATLAB on their own. An additional benefit is that MATLAB reinforces the concepts taught in C such as loops, indexing, conditionals, inputs, storage management, data and program structures, etc. Learning MATLAB within a C programming course provides incentives and variety since MATLAB help create visual effects and animation for simple projects in such basic courses.

Students use a MATLAB textbook[16] for this course and keep as a reference in follow-on courses. Instructors in those courses can introduce toolboxes and provide handouts for their usage without loss in course time to teach MATLAB scripts and functions. Having to learn another high level programming (scripting) language along with C cannot be too significant an extra effort when both are being taught provided the course content is managed properly. A survey was done at the end of the semester since fall of 2012. Regular grading assessments also included MATLAB specific problems to keep students motivated in their learning. In projects toward end of the course, students demonstrated their grasp of programming constructs and also creativity in visual graphics.

Part II of the paper provides a summary of this programming course objectives and the content; Part III lists the specific project assignments done in C and MATLAB over several years;   Part IV ponders upon the results of surveys answered by students in this and upper level follow-on courses to determine the added value of MATLAB as a learning tool; and Part V states concluding remarks.

Part II: course content and objectives

The objective of the course and the assignments were to provide students with a skill set so they would be able to analyze the broader picture, identify the actual problem, develop a solution, design and implement a software program for that solution. Students are to use structured forms in dividing the solution into distinct and coherent functions with data exchanged via parameters. Those actions are straight forward in both C and MATLAB. However, MATLAB allows multiple result to be passed back directly whereas one has to use data pointers (arrays) in C to pass back multiple values.

Table 1 below lists the course topics and objectives while stating whether MATLAB was discussed with regard to that topic or not. Whenever MATLAB was contrasted, its usage in writing code was assessed (in most cases) in a homework, a quiz, or in an exam to a lesser extent than its C counterpart. Author kept the rigor of MATLAB tasks to a degree less than that of C tasks in order to ensure the major emphasis was on C as the formal requirement of the course. Topics or their depths were not allowed to diminish by the added MATLAB content.

The course was not partitioned into C and MATLAB, but they were intertwined at each programming topic if deemed necessary. The course mainly followed the topics in the C textbook[14] while MATLAB textbook[16] was used to compare and contrast relevant topics. Every C topic was not countered with contrasting MATLAB features since some may not have a difference except a triviality (example: printf); MATLAB feature may not be relevant for them (binary bit wise operations); or the feature was missing (e.g. data pointers).

	Topic	Course Objective
1	Computing machines	Understand basic arithmetic and logic units along with speeds of different memory types.
2	Number Systems	Understand binary, hexadecimal, decimal number systems on positive and negative values.
3	Data Types	Understand several C language data types, usage, and conversions
4	Operations	Understand arithmetic operators, basic input-output operations, and C language intrinsic functions; compare and contrast with Matlab.
5	Decisions and Control	Understand C operators and their use in implementing structured program control structures; compare and contrast with Matlab.
6	Explicit operators	Understand bitwise and logical operators; compare and contrast with Matlab
7	File Access	Understand basics of file read and write ; compare and contrast with Matlab.
8	Modular Programming & Functions	Understand and master the concepts of C modules and functions, their structures & activation, recursion, variable and function scopes, data persistence, etc.
9	Arrays	Understand structure of single and multidimensional arrays, indexing, table lookup; compare and contrast with Matlab.
10	Pointers	Understand concept of pointer variables, usage, function/data access
11	Structures	Introduction to data structures, typedef, exposure to simple examples (array of data records, singly linked lists & insert/remove, sorted lists)
12	Program Design and Implementation	Demonstrate above by the design and implementation of C language programs in a contemporary Integrated Development Environment (IDE) – compile, build, debug tasks; compare and contrast with Matlab environment

Table 1: Course Objectives in Computer Systems, Programming, and Applications in C

Students are made aware of interpreted nature (JAVA, MATLAB, etc) and compiled nature (C, C++, FORTRAN) of processing software code by underline systems. The contrast between C and MATLAB widens with "explicit" operators, array handling, and functions. They feel more comfortable in C functions - perhaps due to the fact it was the major emphasis in the course, but some find it difficult to grasp the idea of returning values to a caller in C due to its abstract nature (in MATLAB a variable is assigned the values that are to be returned). In order to reinforce generic programming constructs, the use of array based operations and built-in functions were discouraged though mentioned in class.

Pointer concept is very new to them, but most of them grasp the concept gradually after several examples and demonstrations. It is noted that MATLAB does not offer a concept of a pointer to its users which limits its efficiency in writing very large software applications for complex problems due to memory storage management issues which have been discussed in on line user community. MATLAB has its own dynamic binding and garbage collection mechanism which make it a bit slower in run time and less efficient in memory use.

In using dynamic storage such as arrays with changing sizes, MATLAB users do not explicitly write commands to allocate/free data storage unlike C/C++ programmers. The memory management is implemented by MATLAB software[15]. However, internally it uses access by reference when arguments are passed to functions which is in fact pointers to original data variables. As long as the input data is not modified by the function being called, the variable in the calling function and the variable in the called function point to the same data location in memory. If the function changes the variable value, a copy is made to avoid changing original data. Demand for MATLAB workspace memory may peak temporally if a very large array is expanded to add more data since it has to first allocate a larger contiguous memory block to accommodate the larger one before copying the original. In this exchange, it may slow down the programs unexpectedly or run out of memory depending on workspace size and computer resources. In contrast, C code writer manages storage explicitly by allocating and reallocating at appropriate locations and has much more flexibility in memory efficiency.

In this course, the major attraction of MATLAB over C is in visualization as evident from the animation project described in Part III. Students were exposed to C language storage management techniques in the spell checker project outlined in Part III below. The major difficulty students seemed to have had was switching between the syntax of C and MATLAB, particularly in the area of *arrays* and *for* loops. Rather than making item 12 in Table 1 a single topic, it was taught hands on throughout the semester as the need and opportunity arose.

Part III: course projects

Homework typically reflects on the material covered in the lecture class. For any portion of assignments that involve coding, students use DevC/C++ and MATLAB on Microsoft Windows based desktop computers in computer labs. They develop the algorithms in most cases, and only when the problem was deemed difficult for their level, the author provided a skeleton of the algorithm or the solution. Students were provided with example code segments and hands on practice code for certain topics in order to drive the ideas home.

About six or seven weeks into the course, after students have practiced writing basic loops, case statements, arrays, strings, formatting and input/output functions, they were assigned projects in C programming. Early revelation of project assignments make students aware of the seriousness of the course and allows ample time to work on them. Students were allowed to form groups of two or work alone on projects. Most of them formed groups rather than work alone and that arrangement worked well. Their deliverables are a typical flow chart or a description as phase I, a working program submitted via course web or blackboard, and most critical input values to use their program successfully if the project were a computer game.

Since DevC++ does not offer visualization and graphical manipulations, a MATLAB based project was also given to them. These projects were to animate the movement of drawn objects on a fixed graphical window using *plot* function along with delays without use of special Toolboxes which would have required a much greater effort to learn than the time permitted. Basic MATLAB functions were sufficient to make student practice and feel that they can have fun while learning. MATLAB is very user friendly for data visualization purpose, although other software packages are better suited for commercial computer animations. The animation was achieved by repeated plot command applied on the same data set (which draws figures), but shifted by (Xo,Yo) coordinates, where (Xo,Yo) represents the new location of the figure. This can be duplicated into any number of moving figures by having different figures and different (Xo, Yo) locations. In order for human eye to see the movement smoothly, the flicker rate needs to be more than 24 frames per second. The visible delays are easily achieved using *pause* command in Matlab. The projects are to be designed with the limited time available to learn basics of programming and command syntax.

Two projects were assigned every year – one in C and another in MATLAB showing some type of animation which is more exciting than mere textual outputs. Prior year projects were not repeated to avoid the use of previous solutions, and the author made it known that any project group may be quizzed on the code they submit for grading to ensure that third parties were not providing the solutions. Grading was significant that about 10-12% of final grade came from the two projects.

C projects:

(i) *A calculator project to implement a subset of the functions available in the standard Windows based desktop PC calculator*: It is a software tool without a graphical user interface. The actual calculators on Window based systems use mouse and keyboard functions inside a Win32 programming environment. However, the course is the very first programming course and learning Win32 system programming at this level is neither feasible nor necessary. Students utilized arrays, switch statements, and other constructs to manipulate data and arithmetic operators after reading them from the keyboard.

(ii) *A spell checker program to read in a text based large dictionary file and to compare words in another text file against the stored dictionary words*: It is a software tool without a graphical user interface. The dictionary is initialized by reading in a text file containing the words, one per line and storing them in a global string array. The program reads in a text file as the document to be checked for spelling, and writes out the misspelled words and a summary of statistics including specific run times. When the same dictionary file is used as the document, there should not be any misspelled words. The project consisted of an enhance phase to be implemented to speed up the checking process after students get it working correctly. This reinforces the notion of finding multi tier solutions to achieve efficient results, and exposes students to dynamic memory management concepts in programming languages.

(iii) *A reverse polish notation (RPN) based calculator project to practice and learn array and stack manipulations*: It is a software tool without a graphical user interface. A standard calculation sequence is to provide the operation in the middle of the expression where as RPN provides the operation after the expression (postfix) which makes the expression unambiguous and removes the need for parenthesis. This allows a few more operators beyond +, -, /, etc, in order to control and view the sequence of information received before the final value is calculated by the tool. The following control operations were specified for implementation. Data is to be entered using the keyboard or reading a text file.

Operator character	Function
C or c	Clear last entry
D or d	Dump entire sequence of entries
E or e	Erase everything given so far
=	Calculate and give final value
+ - / * % ^	They have the same meaning as in C language
@	Raise to the power, example: 2 3 @ gives 8

MATLAB projects:

(i) The project was to draw (plot command) an air plane, balloon, helicopter, bird, or superman, or any flying object on a fixed size figure (graph) window. It was to be smooth looking with sufficient number of data points and properly scaled. Students were provided with an example animation of an analog clock and a walking man as a start. In order for them to have a game-like feel, a projectile was added as a second moving figure which tried to intercept the other moving object. User was given the control of the projectile parameters such as velocity and the launch angle. The interception was conveyed by indicating an explosion (a flicker of yet another figure for a few seconds) if the two moving objects came visually close enough as calculated by the coordinate distances.

(ii) This was to create few different animated flying objects in a graph window along with a colorful landscape using additional commands. As with the previous project, a moving projectile was added which would try to hit one or more flying objects or moving objects on the landscape grounds. More points were given if they would code in higher achievement levels if the player is successful in hitting the moving object. Students were directed to tryout NLOS cannon challenge game on line to have an idea how video games are animated. Som eof the students were very creative in animation and color coordinations. It was noteworthy that one student implemented all 15 levels just as done in that game with various animated objects.

(iii) This animation project in MATLAB was to move different size balloons - at least three in random (but smooth flying) directions. Somewhere a circle or a rectangle with sharp edges is to be placed which takes about 10% of plot area. When the flying objects hit it, they were to burst and disappear. That destruction was also to be animated by showing some level of creativity. Moving objects should show random speeds but the same general direction (up, down, left, right, etc) when moving from one edge to the other of the plot window. Movement may be enhanced using random decision making to change the direction and speed of objects after collision with another object or a window edge.

Part IV: student feedback

Since the first introduction of MATLAB to this course, students were surveyed every year at the end of the course to gauge learning effectiveness of having MATLAB as an additional programming environment in that course. While they would reap the benefit of having learned C language immediately in several courses at junior and senior level, knowing MATLAB in the current semester would help them immediately in the following semester when they need to use MATLAB for the Circuits II class. Same student group was surveyed at end of Circuits II course and again surveyed two years later after they have taken the Control Methods course. The survey scheduled is shown in Table 2. The responses were given based on the typical rubrics of 1 to 5 (1-strongly disagree, 2-disagree, 3-neutral, 4-agree, 5-strongly agree). In Table 2 column headings, average number of students surveyed is given in parenthesis for each course. These courses have small class sizes - typically less than 20.

Student group	C + MATLAB (18)	Circuits II (15)	Control Methods (15)
1	Fall 2012	Spring 2013	Fall 2014
2	Fall 2013	Spring 2014	Fall 2015
3	Fall 2014	Spring 2015	-

Table 2: Student Surveys Conducted

Typically only electrical and computer majors take the C programming course. There is only one course section every year. Therefore, it was not possible to have a parallel control group to gather data for comparison. The prior year students had already used MATLAB in a spring semester course many months before the author contemplated this change in 2012. Therefore, attempting to gather meaningful survey data from that group to compare with first group in 2012 fall would not have been fruitful.

The survey results for C + MATLAB course are summarized qualitatively in Table 3. Column 2 of Table 2 lists what each question was trying to assess from students' perception and knowledge. The 3rd column shows the average rubric score given by students for each question. Final column in Table 2 attempts to give a qualitative meaning to the data in column 3 so that author can make qualitative adjustments to the course content and projects in the future.

Almost all the questions posed to students received an affirmative answer of "yes" in the qualitative score indicating that the course content is adequate for them to learn programming while keeping the emphasis on learning C- language rather than MATLAB commands. Item 5 indicating "no" is a form of a validation that by placing emphasis on C (over MATLAB) students' preferences can be channeled in the beneficial direction. Even though the student body is relatively small, qualitative answer column turned out to be consistent over the three surveys. .

	Essence of the question asked	Average	Qualitative
1	Students learned C programming well in this course	3.8	yes
2	C based project helped students learn C programming in this course	3.6	yes

3	The moving objects and animation project helped students learn MATLAB in this	3.5	yes
4	C - programming done was challenging (made me think) but it was fun	3.8	yes
5	Students preferred MATLAB programming over DevC/C++ programming	2.2	no
6	Students like to think in details, hence C and MATLAB programming are OK.	3.6	yes

Table 3: Average Survey Results in C + MATLAB Course

Table 4 provides the average results of three surveys (see Table 2) done at the end of Circuit II course which students take right after C programming course. Students initially may feel MATLAB as extra work in a C programming class. After they have used what they learned, it becomes appreciated and accepted as question 2 answer show. Question 3 and 4 answers show that students learned some of MATLAB they needed or helped learning more in C programming course since everything was not learned in the follow-on course. Those responses are consistent with the results of question 5 although adding more non C topics may hamper the main objective of learning C programming. Next question reveals that students still remember - after 4 months - that they were compelled to learn MATLAB in a C course. Their perception changed after using what they felt like they were forced to learn in the prior course – from average response of 3.7 went down to 2.2 showing a moderate disagreement toward the notion that it was extra work. This is probably true of any material that they would be compelled to learn. After they see the direct benefit of learning something - because they were compelled to learn it - the notion that it was forced upon and perception of additional burden slowly disappear.

	Essence of the question asked	Average	Qualitative
1	Students still think they learned C programming well in C+MATLAB course	4.2	yes
2	MATLAB should not go along with C in the programming course	2.1	No
3	Student learned all required MATLAB in Circuits II course itself	1.8	No
4	Learning MATLAB basics in C programming course helped me learn advanced features easily in follow-up classes	3.7	Yes
5	Students wish they had learned more MATAB in C programming course	3.8	Yes

6	Students viewed learning MATLAB as extra work at beginning in C programming course	3.7	Yes
7	Students still consider learning MATLAB in C programming course as extra work	2.2	No

Table 4: Average Survey Results in Circuits II Course

Table 5 provides the average results of two surveys done at the end of Control Methods course taken at the end of first semester of senior year by both electrical and computer technology students. They took the C + MATLAB programming course at least 2 calendar years ago and used MATLAB in Circuits II course and were just finishing up the control course. Survey questions somewhat resembles to those in Table 4 since the idea is to gauge how they benefitted and what they think of MATLAB piggybacking on C course now that they have almost completed using MATLAB in the undergraduate curriculum. They may still be working on senior projects with C programming though.

Responses to first couple of questions are similar to those in Table 4 as expected. Unlike Circuits II course, students need to learn MATLAB toolbox based advanced commands and Simulink which are introduced in laboratory sessions in the control class yielding answer to questions 3 and 4. Question 5 validates that they have received sufficient level of C programming breadth and depth to work and solve problems in follow-up programming courses such as Data Structures for computer students and microcontroller programming in Embedded Systems course which use C language extensively for lab sessions and projects.

Question 6 in both Table 4 and Table 5 poses the same question and received similar responses as expected. Even with in the stated objectives of a course, students may feel that certain in–depth topics are unnecessary and extra work. It was obvious that students would consider learning MATLAB as extra work since course title stated only C programming. However, a slight reduction of that perception is observed in Table 5 compared to Table 4. The last question in Table 5 reveals that students still remember that they were compelled to learn MATLAB in a C course, but their perception has changed after using what they felt like they were forced to learn. This is probably the validation of this endeavor of adding extra burden to students as well as to the teaching faculty with the introduction of MATLAB programming in a C programming course without diluting the C course.

	Essence of the question asked	Average	Qualitative
1	Students still feel they learned C programming well in C + MATLAB course 2 years ago.	4.1	yes
2	MATLAB should not go along with C in the programming course	2.1	no
3	Students learned all required advanced MATLAB features for Control Methods course in class itself	4.0	Yes

4	Learning MATLAB basics in C programming course helped me learn advanced features easily in follow-up classes	3.9	Yes
5	I do not feel learning MATLAB reduced topics or learning C in C programming course.	3.8	Yes
6	Students viewed learning MATLAB as extra work at beginning in C+MATLAB course	3.5	Yes
7	Students still consider learning MATLAB in C programming course as extra work	2.1	No

Table 5: Average Survey Results in Control Methods Course

Part V: conclusion

The decision to introduce MATLAB in C-programming course at this university campus was made for the fall 2012 semester. It was done with ample caution and care so as not to overload students having to learn MATLAB in addition to the already established study load in C programming course. Learning C cannot be scaled back due to its necessity and that being the only formal programming course in the degree programs. As the initial student survey indicated in 2012, the two competing programming tools were balanced right and this experiment will encourage the continuation of teaching MATLAB in C course by the author. That data was presented in a previous conference paper[3] by the author.

This paper discusses the continuing experience of students in upper level relevant courses after learning MATLAB as an additional programming tool at the sophomore level. The objective of adding extra learning was to reduce their future burden of learning MATLAB on their own while taking courses with challenging and complex mathematical concepts. They still learned advanced concepts and toolboxes in higher level courses. They had learned MATLAB basics and it would be easier to build on what they already know. It allows more time for actual subject matter in upper level courses.

In conclusion, the incorporation of a certain amount of MATLAB into an introductory C-programming course brings both immediate and near future benefits for both students and instructors. Students enjoy the visualization and rich graphical interfacing functions that MATLAB offers as revealed by three animation projects described in here while strengthening their programming skills learned in C language. As a result, they will be more comfortable in learning other features of MATLAB such as Control Toolbox usage for their follow on courses. Instructors benefit by having students who have already been exposed to basic MATLAB and hence can easily be stepped up to the next level of required functionality without allocating class or lab time for introductions.

Results of surveys done in upper level courses at different times in their curriculum validate that this endeavor was a success and should continue. In the long term, students will benefit by having gone through several follow on semesters learning more MATLAB on their own which would instill the practice of lifelong learning making them successful electrical and computer engineering graduates.

References:

[1] Qin, Qing-Hua, Wang, Hui, "Matlab and C Programming for Treffitz Finite Element Methods," Taylor & Francis Publishing, July, 2008.

[2] "The Right Tool for the Job," http://hammerprinciple.com/therighttool/items/matlab/c 2012

[3] Karunaratne, M., "Learn MATLAB Piggybacked onto C-Programming," Proceedings of ASEE Annual Conference and Exposition, June 2013, Atlanta, GA.

[4] Frontoni, E., Mancini, A., Caponetti, F., Zingaretti, P., "A framework for simulations and tests of mobile robotics tasks," Control and Automation, 2006. MED '06, The 14th Mediterranean Conference on Control and Automation.

[5] Nelson, M.L., Rice, D., "Introduction To Algorithms And Problem Solving," Proceedings 2000 Frontiers in Education Conference, Oct. 2001, Kansas City, MO.

[6] Raymond, D.R., Welch, D.J., "Integrating Information Technology And Programming In A Freshmen Computer Science Course," Proceedings 2000 Frontiers in Education Conference, Oct. 2001, Kansas City, MO.

[7] Ludi, S. Collofello, J., "An analysis Of The Gap Between The Knowledge And Skills Learned In Academic Software Engineering Course Projects And Those Required In Real Projects," Proceedings 2001 Frontiers in Education Conference, Oct. 2001, Reno, NV.

[8] Wong, P., & Pejcinovic, B., "Teaching MATLAB and C Programming in First-year Electrical Engineering Courses Using a Data Acquisition Device," Proceedings ASEE Annual Conference and Exposition, June 2015, Seattle, Washington.

[9] Ageenko, E., La Russa, G., "A Visualization Toolkit for Teaching, Learning and Experimentation in Image Processing," Frontiers in Education Conference 2005.

[10] Aleksi, I.; Kraus, D.; Hocenski, Z., "Multi-language programming environment for C++ implementation of SONAR signal processing by linking with MATLAB External Interface and FFTW," Proceedings ELMAR 2011 Conference, September 2011, Zadar, Croatia.

[11] Fincher, S., "What are We Doing When We Teach Programming?," Proceedings 1999 Frontiers in Education Conference, No. 1999, San Juan, PR,

[12] Jermann, W.,H., "The Freshman Programming Course: A New Direction", Proceedings 1996 ASEE Annual Conference, June 1996, Washington, DC.

[13] Westbrook, D.S., "A Multiparadigm Language Approach to Teaching Principles of Programming Languages", Proceedings 1999 Frontiers in Education Conference, Nov. 1999, San Juan, PR

[14] "Programming in C," 4th edition by Stephen G. Kochan, ISBN-10: 0-672-32666-3, Sams Publishing 2004

[15] The Mathworks, "How MATLAB Allocates Memory," on line technical document at http://www.mathworks.com/help/matlab/matlab_prog/memory-allocation.html

[16] "Engineering Computation with MATLAB," 3rd edition by David Smith, ISBN-10:0132568705, Pearson Education 2012.