Electrical Engineering

Response to ABET EAC Final Statement of August 13, 2007: Criterion 2 Weakness

June 20, 2008

White paper submitted in response to United States Air Force Broad Agency Announcement 02-06-IFKA, Amendment 2.

Key Contacts:

Reza Abbaschian, Dean, Bourns College of Engineering
(951) 827-5190, rabba@engr.ucr.edu

Roger Lake, Chairman, Department of Electrical Engineering
(951) 827-2122, rlake@ee.ucr.edu

Amit K. Roy-Chowdhury, Department of Electrical Engineering
(951) 827-7886, amitrc@ee.ucr.edu

Mitch Boretz, Office of the Dean
(951) 827-7069, mitch@engr.ucr.edu

Electrical Engineering response to ABET 2008.

The document was prepared by Dr. Amit K. Roy-Chowdhury, Dr Roger Lake and Mr. Mitch Boretz, with input from all faculty in the Electrical Engineering Department.

Table of Contents

1.0 Introduction

The 2007 ABET review contained the following weakness:

> **Criterion 2. Program Educational Objectives** Criterion 2 states that the program must have "a process based on the needs of the program's various constituencies in which the objectives are determined and periodically evaluated." While a process exists, it is not clear that this process is clearly tied to feedback from the program's defined constituencies or what the time period is for re-evaluations of these objectives. Some objectives appear difficult to measure and some are similar to outcomes. Criterion 2 states that the program must have "a process of ongoing evaluation of the extent to which these objectives are attained, the result of which shall be used to develop and improve the program outcomes so that graduates are better prepared to attain these objectives." (While Evaluation has been done, it is not yet clear that this is an ongoing process and that the loop is being closed to use in the evaluation results in program improvement.)
>
> - Due-process response: The EAC acknowledges receipt of the documentation describing activity to define a new set of educational objectives that are focused on early career accomplishments and a description of a process for involving constituents in developing and refining the objectives. The target date for approval of the new objectives and objective review/definition process was indicated to be May 2007.
>
> 1) The weakness remains unresolved and will be the focus of the next review. In preparation for the review, the EAC anticipates evidence documenting the implementation of the new process.

2.0 Response

As noted in the due process response, the weakness was being actively addressed by the department in the Winter and Spring of 2007. This document demonstrates that the weakness has been thoroughly addressed. New Program Educational Objectives (PEOs) have been approved by all stakeholders. A well-defined, closed-loop evaluation and program-improvement process is in place. The details are presented below.

A new set of PEOs was adopted (see Table 1 below) and approved by the faculty, board of industrial advisors and students. The process of revising the PEOs involved extensive discussions within the department's Assessment and Accreditation committee (consisting of four faculty members, the department chair and a staff representative); obtaining inputs from faculty, students, employers and alumni; and final approval by all the constituents. Minutes of the faculty meeting are attached in Appendix B. Minutes of the board of advisors (BOA) meeting in 2007, which approved the PEOs, are attached in Appendix C. Comparison of the PEOs in Appendices B and C will show that they evolved through inputs from the faculty and BOA – an initial set of revised PEOs was developed with input from the faculty, which was then placed before the BOA and student groups, which gave their own input, and the final set of PEOs was then again presented before the faculty for its approval. Minutes of the BOA meeting in 2008 where the PEOs were revisited and analyzed is attached in Appendix D. Approval from student groups is attached in Appendix E.

We have implemented a process for evaluating the success of the PEOs through regular surveys of our alumni and industry representatives, who are the employers of our former students. The questionnaires prepared for surveying the alumni and employers are attached in Appendices F and G, respectively. This survey will be conducted annually. It is done by sending an email invitation from the department chair to all of our alumni and their employers. A database for this purpose is being maintained at the college level. In addition, the chair of the Electrical Engineering Assessment and Accreditation Committee will contact student groups on campus (specifically the student chapter of IEEE and the College of Engineering Leadership Council) for their comments on the PEOs.

Table 1: The New Program Educational Objectives

Graduates of UCR's BS degree program in Electrical Engineering will meet high professional, ethical, and societal goals as demonstrated by:

success in *post-graduation studies as evidenced by*:

- satisfaction with the decision to further their education
- advanced degrees earned
- professional visibility (e.g., publications, presentations, patents, inventions, awards)
- professional responsibilities (e.g. professional mentoring, professional society membership and offices, reviewing and editorial work for professional journals)

success in *a chosen profession or vocation as evidenced by*:

- career satisfaction
- promotions/raises (e.g. management leadership positions or distinguished technical positions)
- professional visibility (e.g., publications, presentations, patents, inventions, awards)
- professional responsibilities (e.g. professional registration, professional mentoring, professional society membership and offices)
- entrepreneurial activities
- consulting activities

contributions to society as evidenced by:

- leadership roles
- public service
- mentoring / outreach activities
- volunteer service

Based on these sources of feedback, we have implemented a process of improving our program so as to better attain the objectives. The process for this implementation is as follows. Once the survey results come in, the department committee on Assessment and Accreditation reviews the results and suggestions and, as appropriate, recommends a set of changes. The committee works with the undergraduate committee to develop the best method for implementing the changes. These are then brought forward for discussion at a departmental faculty meeting. Feedback from our Board of Advisors, consisting of representatives from industry, is also obtained. Once faculty approval is obtained, the implementation process starts. Depending upon the nature of the change, the process may require approval by the Academic Senate of the university. Documentation of all the changes made to our program and how they affect the PEOs is being maintained.

To illustrate that we are currently implementing this process, we provide an example of a major change – introduction of a programming course that combines data structures, algorithms and numerical computation. Employers provided an input to the Department Chair that students were inadequately prepared in programming skills. The undergraduate advisor also had multiple informal interviews and meetings with senior students and students who did summer internships and National Science Foundation Research Experiences for Undergraduates (REU) programs. The feedback from the students also indicated strongly that they needed more training in programming. The department's board of advisors also indicated the need for strengthening this aspect of the curriculum. The undergraduate committee evaluated the curriculum and decided to add a second programming course specialized for EE students. To make room for this new course, the undergraduate committee suggested that ME10, Statics, is not used in most if not all EE professions. The undergraduate advisor consulted with industry contacts and confirmed this. The undergraduate committee presented the findings to the faculty and recommended that ME10 be removed and CS13 Programming for EE be added as a required course. The faculty voted unanimously to approve the change. The undergraduate committee has received inputs from EE faculty, EE advisory board and industry contacts, and is working with the CSE department to develop the syllabus for CS13 based on these inputs.

To ensure that this is a continuous process that is implemented formally within the department, we have set up a process for surveying our alumni and their employers every year and implementing the recommended changes. The surveys are emailed out early each calendar year by the Department. Once the surveys come back, the Assessment and Accreditation (AA) committee will evaluate the feedback from the surveys and then present any changes to the Undergraduate committee (UG) by late Winter. A set of recommended changes is proposed by the AA and UG committees in the Spring and sent to the faculty, board of advisors and student groups for their comments. The final set of recommendations is voted on by the faculty by the end of the Fall quarter. After that, the approved changes are sent for approval at the College and campus level. If approved, the changes will appear in the following Fall's catalog. This will be an annual cycle. The evaluation of the feedback will be coordinated by the chair of the Assessment and Accreditation committee, who will interact with the chair of the Undergraduate committee. The Department chair will coordinate the overall process. The timeline for this process and the associated responsibilities are shown in Fig. 1.

A compressed schedule was followed this year because we had to prepare the questionnaires over the Winter quarter and the surveys were conducted in the Spring. This is the first year when we are conducting such a large-scale survey. We have received about 40 responses. Given that we are a young college, we believe that the number of responses is reasonable for the first year. Some of the issues raised will be addressed as described above. We are also working within the College and with the relevant campus offices on obtaining more reliable contacts of our alumni which should improve the number of responses.

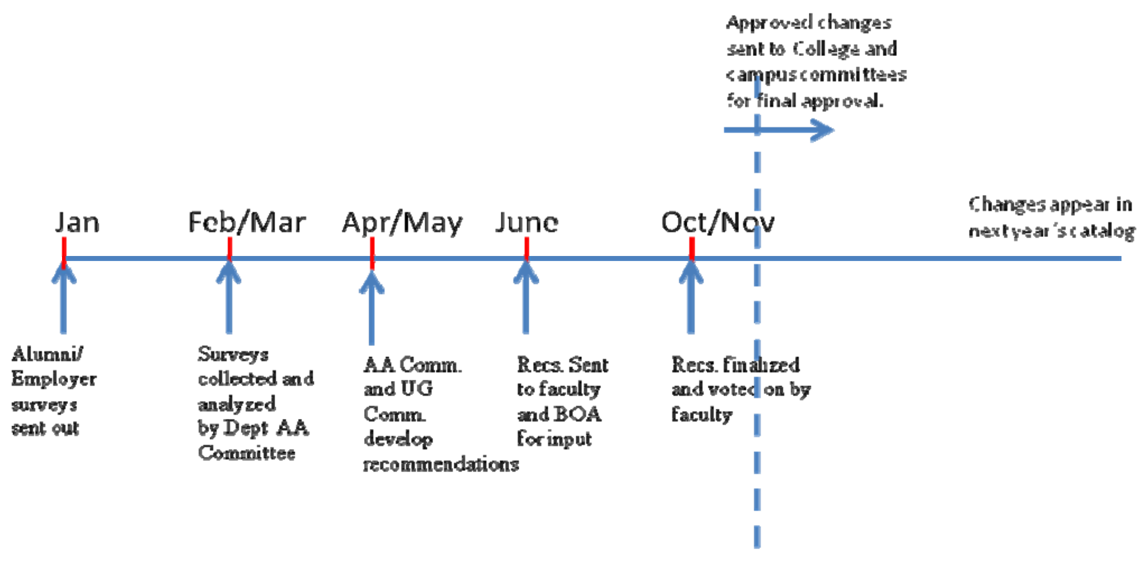


Fig. 1: Timeline and responsibilities of feedback cycle based on evaluation of the PEOs.

3.0 Additional Information on Program Outcomes and Assessment

Criterion 3, Program Outcomes and Assessment, was cited as a concern. While we understand that we do not need to respond to a concern, we do provide some documentation about the changes that were effected as per our response in 2007. The following was the text from ABET regarding Criterion 3.

> **Criterion 3. Program Outcomes and Assessment** Criterion 3 requires "… an assessment process, with documented results, that demonstrates… program outcomes are being measured and indicates the degree to which the outcomes are achieved." While some assessment has been implemented, it does not appear that all outcomes are sufficiently measured and that achievement of all outcomes is being demonstrated. Sufficient evidence was not provided for the following

outcomes: "b" and ability to design and conduct experiments, "d" an ability to function on multi-disciplinary teams, "f" an understanding of professional and ethical responsibility, "h" the broad education necessary to understand the impact of engineering solutions in a global, economic, environmental, and societal context, "j" a knowledge of contemporary issues."

- Due-process response: The EAC acknowledges receipt of documentation identifying curricular changes that have been implemented to insure adequate coverage of and documentation of achievement of Criterion 3 items b, d ,f and j. Because the program relies heavily on grading as an assessment tool, outcomes b, d, f and j now appear to be covered within the curriculum so that better assessment will take place.
- The weakness is now cited as a concern pending demonstration of the robustness of the changes.

To recap, we had stated the following in our Response to Draft ABET Findings regarding Criterion 3.

"*While items b, f, h, and j have been part of the material taught in the two quarter Senior Design project, EE175a,b, the demonstration of these outcomes was not well documented. The syllabus of EE175a,b has been rewritten so that these outcomes are explicitly included with corresponding assignments that will be documented and used to measure the students' performance in obtaining these outcomes. The revised syllabus explicitly showing the assignments associated with these outcomes is shown in Appendix A. The final report template with required sections related to these areas is shown in Appendix B. Starting this Winter and Spring quarters, items b, f, h, and j will be documented for all EE students in either the EE175 final report, exam, or essay assignment. These instruments will be used to measure the program outcomes*

.

Item d, an ability to function on multi-disciplinary teams, is being addressed in the two cross-listed courses EE/CSE120A/B. All Electrical Engineering, Computer Engineering, and Computer Science students are required to take this sequence of courses. It was decided that for both EE/CSE120A and 120B, lab partners will be rotated weekly until every student in the class has worked on one interdisciplinary team. The lab assignments from the interdisciplinary teams will be an instrument for demonstrating and measuring the students' ability to function on interdisciplinary teams. The revised catalog text for the courses is shown in Appendix C, and the revised syllabus for 120A is shown in Appendix D."

As noted in the due process response, the weakness was addressed by the department in the Winter and Spring of 2007. Here we document the continuation of this process and demonstrate the robustness of the changes. Appendix H shows the instructions for EE175A/B, which show the different sections highlighting items b, f, h and j. Three samples (best, median and poor) of reports from students in the 2007-2008 academic year are presented in Appendix J. These will demonstrate that the changes we proposed have been fully implemented and are part of an ongoing process. Based on student feedback, one improvement implemented this year was the following: "Started the project selection process in the Fall quarter of 2007 so that students have a project ready to start at the beginning of Winter quarter 2008." – Prof. Ping Liang, Lead Instructor for EE175.

In Appendix I, we provide a description from the instructor of EE120A/B on how the multi-disciplinary teams were implemented in the 2007-2008 academic year. This also demonstrates the continuation of the process we outlined in the due process response.

Appendix A: The Old Program Educational Objectives

… to produce graduates who shall be able to:

PEO 1. *develop and pursue successful careers in electrical engineering (EE)*

PEO 2. *apply EE knowledge and skills to further careers in a broad range of professional occupations*

PEO 3. *conduct successful graduate studies and research at major research universities*

PEO 4. *demonstrate innovation and creativity, and pursue lifelong learning in solving engineering problems*

PEO 5. *work effectively in a team environment, communicate well, attain professional growth, and provide leadership in engineering*

PEO 6. *exercise professional responsibility and sensitivity to a broad range of social concerns, such as ethical, environmental, economic, regulatory, and global issues*

Appendix B: Electrical Engineering Faculty Meeting Minutes regarding PEOs

Feb. 14, 2007 (the first draft of the new PEOs was discussed and approved)

Attendees/Faculty: Roger Lake, Ilya Dumer, Sheldon Tan, Ertem Tuncel, Amit Roy-Chowdhury, Alex Korotkov, Jianlin Liu, Matt Barth, Jie Chen, Jay Farrell, Alex Balandin, Ping Liang
Attendee/Staff: Mark Bourbonnais

1) Discussion and vote on revision to Program Educational Objectives
   a) Program educational objectives to be amended as follows
      Graduates of UCR's BS degree program in Computer (Electrical) Engineering will be capable of achieving:
      success in post-graduation studies as evidenced by:
      - satisfaction with the decision to further their education
      - advanced degrees earned
      - professional visibility (e.g., publications, presentations, patents, inventions, awards)
      - international activities (e.g., participation in international conferences, collaborative research, employment abroad)

      success in their chosen profession as evidenced by:
      - career satisfaction
      - promotions/raises
      - professional visibility (e.g., publications, presentations, patents, inventions, awards)
      - entrepreneurial activities
      - international activities (e.g., participation in international conferences, collaborative research, employment abroad)

      During discussion it was suggested to remove "international activities" and move the "participation in international conferences" to "participation in conferences" under professional visibility. The PEOs below were unanimously approved to be sent out to our other stakeholders for comments and suggestions.

      Graduates of UCR's BS degree program in Computer (Electrical) Engineering will be capable of achieving:
      success in post-graduation studies as evidenced by:
      - satisfaction with the decision to further their education
      - advanced degrees earned
      - professional visibility (e.g., publications, presentations, participation in conferences, patents, inventions, awards)

      success in their chosen profession as evidenced by:
      - career satisfaction

- promotions/raises
- professional visibility (e.g., publications, presentations, participation in conferences, patents, inventions, awards)
- entrepreneurial activities

May 31, 2007 (New PEOs, with BOA and student input, were discussed and approved)

Attendees/Faculty: Roger Lake, Amit Roy-Chowdhury, Ertem Tuncel, Sheldon Tan, Jianlin Liu, Jay Farrell, Ilya Dumer, Yingbo Hua, Sakhrat Khizroev, Alex Korotkov, Alex Balandin, Daniel Xu.

Attendee/Staff: Vanda Yamaguchi

The following set of ABET PEOs were approved after incorporating changes from BOA.

Graduates of UCR's BS degree program in Electrical Engineering will meet high professional, ethical, and societal goals as demonstrated by:

success in *post-graduation studies as evidenced by*:

- satisfaction with the decision to further their education
- advanced degrees earned
- professional visibility (e.g., publications, presentations, patents, inventions, awards)
- professional responsibilities (e.g. professional mentoring, professional society membership and offices, reviewing and editorial work for professional journals)

success in *a chosen profession or vocation as evidenced by*:

- career satisfaction
- promotions/raises (e.g. management leadership positions or distinguished technical positions)
- professional visibility (e.g., publications, presentations, patents, inventions, awards)
- professional responsibilities (e.g. professional registration, professional mentoring, professional society membership and offices)
- entrepreneurial activities
- consulting activities

contributions to society as evidenced by:

- leadership roles
- public service
- mentoring / outreach activities
- volunteer service

Appendix C: Electrical Engineering Board of Advisors Meeting Minutes regarding PEOs in 2007

May 18, 2007

Thursday, 05/17/2007
EE Board of Advisors Meeting

Mitch Boretz conducted a briefing to the board regarding ABET which included:

- What is ABET
- EE ABET Overview
- CE ABET Overview
- Requirements of ABET
- Overview of recent ABET review and visit and results
- Explained eight criteria involved with program assessments
- College/Dean's Office role in ABET accreditation
- Boards role in ABET accreditation

Mitch explained the process and the point at which the College is now and the projected outcome in July when ABET will release their official results. To date the feedback from ABET has been positive.

The importance of documentation and objective evidence of improvement was emphasized.

The College proposes having ABET duties as part of the new Placement Director's job.

Mitch explained the board's role in reviewing, commenting and voting on the department's proposed revision to the PEOs.

Mitch gave the floor to Assistant Professor Amit Roy-Chowdhury to conduct the department level ABET briefing.

Professor Amit Roy-Chowdhury conducted a briefing to the board regarding ABET which included:

- Overview of EE and CE process
- Summarized weaknesses identified by ABET reviewer for EE
- Summarized department response
- Reviewed current PEOs
- Reviewed proposed PEOs listed below.

Graduates of UCR's BS degree program in Computer (Electrical) Engineering will be capable of achieving:

success in post-graduation studies as evidenced by:

- satisfaction with the decision to further their education
- advanced degrees earned
- professional visibility (e.g., publications, presentations, participation in conferences, patents, inventions, awards)

success in their chosen profession as evidenced by:

- career satisfaction
- promotions/raises
- professional visibility (e.g., publications, presentations, participation in conferences, patents, inventions, awards)
- entrepreneurial activities

The PEOs were discussed at length and it was decided that rather than vote on the proposed PEOs the board would take further time to review and comment on them. The board was instructed to send comments to Roger Lake and Amit Roy Chowdhury.

A significant portion of the discussion focused on how to include the concept of "service," widely defined, into the PEOs. Specific examples of service given in the discussions were mentoring, someone joining the Peace Corps, someone wiring houses for Habitat for Humanity for 5 years, officer positions in professional societies, and editorships of journals.

As part of this discussion, the category of "Professional Achievements" was suggested, which could include in the parenthetical examples IEEE Section Chair, Professional Mentoring, Professional Registration, Professional Outreach, and Professional Service. Jie Chen put forth the phrase "Professional Responsibility" of which all of the above could be included as examples. The phrase "Professional Responsibility" seemed to meet the approval of all present.

The broader areas of "social responsibility" and "public service" were also brought up.

It was asked how do we define success? Roger Lake responded that the department's goals for graduates of the undergraduate program for 3-5 years after their graduation was for the students to be successful either in (1) their job or in (2) their post-graduate studies. The proposed PEOs simply provided examples that served as quantitative metrics of success in those two categories.

Dennis Rice (board member) noted that the PEOs were generic and could be used by any department. There was discussion of the pros and cons of narrow PEOs specific to EE/CE versus generic/flexible PEOs which can take into account changes in career/academic paths of graduates. Roger Lake noted that since these were objectives to be met by graduates of the undergraduate program 3-5 years after graduation, they must remain generic to include changes in career/academic paths of graduates. He gave as a specific example a student who, after finishing his B.S.E.E. degree, gets a law degree and becomes a patent attorney.

The session was adjourned with a request that the board submit their recommendations to Roger Lake and Amit Roy Chowdhury.

Appendix D: Electrical Engineering Board of Advisors Meeting Minutes regarding PEOs in 2008

Electrical Engineering Board of Advisors
May 14, 2008
ABET/Accreditation/Assessment discussion

Amit Roy-Chowdhury made a presentation on the status of accreditation for the Electrical Engineering and Computer Engineering programs.

The board members suggested that the department provide the board with an annual summary of curriculum changes and the motivation for these changes. Amit and Roger Lake agreed that this would be appropriate.

Ping Liang suggested that we include in the June 2008 submission one or more examples of a curriculum change made based on the process we have outlined.

The board commented that the process we have outlined is “top level” and lacks a degree of detail. They suggested incorporating a timeline, indicating when each activity on the process takes place each year. They also recommended adding names to indicate who is responsible for each step of the process and for closing the loop.

Jean Eassum [board member] suggested that we highlight some of the College-wide undergraduate initiatives, such as the learning communities, which foster interdisciplinary collaboration among engineering students.

Appendix E: PEO Approval from Student Organizations

The PEOs were overwhelmingly approved by the IEEE student chapter on campus. Below is the documentation of the email sent to the IEEE Student Chair for input on the PEOs.

-------- Original Message --------
Subject: Re: Approval of EE PEOs
Date: Mon, 12 Mar 2007 21:00:10 -0700
From: David Keith <dckeith@gmail.com>
To: Amit K. Roy-Chowdhury <amitrc@ee.ucr.edu>
CC: Martin Gawecki <martin.gawecki@gmail.com>

Dear Dr. Roy-Chowdhury,

I brought up the EE Department PEOs at the IEEE General Meeting today and the estimated 50 IEEE members present voted to pass the PEOs as submitted.

Thank you,
David Keith
IEEE Chair 2006-07

On 3/7/07, **Amit K. Roy-Chowdhury** <amitrc@ee.ucr.edu> wrote:

> Dear Martin/David,
> I am the chair of the ABET committee (accreditation program for EE) and I need to get the Program Educational Objectives (PEOs) approved by different constituencies. The EE faculty has approved the following PEOs. Could you please try to have them approved by the student body, especially those from EE? You may also give any comments for future changes. Please let me know when you can get this done.
>
> Graduates of UCR's BS degree program in Computer (Electrical) Engineering will be capable of achieving:
> *success in post-graduation studies as evidenced by*:
>
> - satisfaction with the decision to further their education
> - advanced degrees earned
> - professional visibility (e.g., publications, presentations, participation in conferences, patents, inventions, awards)
>
> *success in their chosen profession as evidenced by*:
>
> - career satisfaction
> - promotions/raises
> - professional visibility (e.g., publications, presentations, participation in conferences, patents, inventions, awards)
> - entrepreneurial activities

Thanks
Amit K. Roy-Chowdhury
--
Assistant Professor
Dept. of Electrical Engineering
University of California
Riverside, CA 92507
Phone: 951-827-7886
Fax: 951-827-2425
http://www.ee.ucr.edu/~amitrc

Appendix F: Questionnaire for Alumni Survey

Subject: Please answer the BCOE Electrical Engineering alumni survey

Dear BCOE alumnus:

Greetings. I hope all is well with you. If you plan to be in the area, I am always on the lookout for alumni, industrial engineers, and academic engineers who would like to share their experience with our undergraduates about life after the B.S. Do not hesitate to contact me.

As part of our efforts to continually improve the educational program in the Bourns College of Engineering, I need ten minutes of your time to complete a short on-line questionnaire about your experiences since graduation. We have designed our degree programs to prepare graduates for success in pursuing higher degrees or for success in starting a professional career. Your comments will help us refine our objectives and methods to assure that a bachelor's degree from the Bourns College of Engineering makes a good start to a promising career.

I also ask that you forward this message to your supervisor, graduate advisor, or human resources department for completion of a brief survey about the employer's perspective.

The survey is brief and anonymous. Your responses are very important to us. If you have any questions or concerns, or if you would like to discuss your undergraduate experience in more detail, please contact Mitch Boretz in the Office of the Dean, (951) 827-7069 or mitch@engr.ucr.edu

I appreciate your continuing involvement with the College and wish you success in your career.

Sincerely,

Roger Lake

Roger Lake
Professor and Chair
rlake@ee.ucr.edu

Electrical Engineering alumni survey:
https://www.surveymonkey.com/s.aspx?sm=4Jps6HvY2eZQfStA9Jerqw_3d_3d

Electrical Engineering employer survey:
https://www.surveymonkey.com/s.aspx?sm=gZDd1tUUu8B63FCD8lnHIg_3d_3d

Electrical Engineering and Computer Engineering alumni survey 2008

1. Please indicate the bachelor's degree you earned at UCR

Electrical Engineering
Computer Engineering
Year

2. Have you pursued or completed any degrees beyond your bachelor's degree in engineering from UCR? Y/N

2a. If completed, please indicate your highest degree completed.
2b. If under way, please indicate the degree you are pursuing.

If you decided to pursue post-graduate education, please answer questions 3-5. If you have not pursued a degree beyond the bachelor's degree, please skip to question 6.

3. Professional visibility

3a. Have you published articles in leading journals and/or made presentations at conferences in your field? Y/N
3b. If so, approximately how many?
3c. Have you been named on any patents or patent applications? Y/N
3d. If so, approximately how many?
3e. Have you received any awards for professional achievement? Y/N
3f. If so, please list some or all of your awards.
3g. Have you engaged in any international research or collaborations? Y/N

4. Professional responsibility

4a. Have you been a program committee member or organizing committee member of a conference in the past three years? Y/N
4b. If so, how many times?
4c. Have you been a reviewer for any journals? Y/N
4d. If so, how many times?

5. Satisfaction with your educational/career path

What is the level of your satisfaction with the overall decision to pursue advanced studies (1-not satisfied   5-highly satisfied)

If you work in addition to pursuing graduate studies, please continue with question 6. If you are a full-time graduate student, please skip to question10.

6. Your career path

From the options below, please choose the one that best describes your current work.

Engineering support
Engineering development
Engineering management
Engineering research
Technical sales/marketing
Business management
Other (explain)

7. Career satisfaction

At this point of your career, what is the level of your satisfaction with your career choice and success in each of the following:

6a. The field you work in (1-not satisfied   5-very satisfied)
6b. The organization you work in (1-not satisfied   5-very satisfied)
6c. Your salary (1-not satisfied   5-very satisfied)
6d. Recognition of your work (e.g., awards within your organization, professional activity, presentations) (1-not satisfied   5-very satisfied)

8. Professional visibility

Have you done any of the following in your professional career since completing the bachelor's degree at UCR?

7a. Had promotions and/or raises? Y/N
7b. If yes, how many times?
7c. Published articles or made presentations in your organization or in your profession? Y/N
7d. If yes, approximately how many times?
7e. Engaged in international activities such as participation in international conferences, collaborative research, or employment abroad? Y/N
7f. Made inventions and/or been listed on patents or patent applications? Y/N
7g. If yes, how many?
7h. Been nominated for any professional or academic awards? Y/N
7i. If yes, how many awards have you won?
7j. Mentored others, either inside or outside your organization? Y/N
7k. Led groups or teams on projects or new initiatives? Y/N

9. Entrepreneurial activity

Have you engaged in any start-up businesses or been involved in any new ventures in your organization? Y/N

<everyone answer>

10. Professional activity.

9a. Are you a member of any professional societies? Y/N
9b If yes, please check all applicable
IEEE
ACM
SAE
what others?
Other (write in)

9c. Have you obtained Professional Engineer certification? Y/N
9d. If no, are you pursuing Professional Engineer certification? Y/N

11. Community and public service. Have you been involved in any of the following activities?

10a. Public service Y/N
10b. Community leadership roles Y/N
10c. Volunteer activities Y/N
10d. Mentorship and outreach activities Y/N
10e. Please describe your activities [short answer space here]

12. Your UCR experience

12a. On a scale of 1-5, how would you rate UCR's impact on your career path (1-Highly negative 5-Highly positive)
12b. How was your preparation at UCR in basic math and science? (Rating 1-5)(Comment)
12c. How was your preparation at UCR in breadth requirements beyond math and science? (Rating 1-5)(Comment)
12d. How was your preparation at UCR in the core curriculum in your major? (Rating 1-5)(Comment)
12e. How was your preparation at UCR in technical electives? (Rating 1-5)(Comment)
12f. How well did your Senior Design Project contribute to your career preparation? (Rating 1-5)(Comment)
12g. If you participated in research as an undergraduate, how well did that prepare you for further pursuits? (Rating 1-5)(Comment)

13. What an engineering graduate needs to know

Our program is designed to enable a Bourns College of Engineering alumnus to be successful either in pursuing a higher degree or in starting a career in engineering or a related field. Based on your experience, what comments do you have on our program and our objectives?

Appendix G: Questionnaire for Employer Survey

Electrical Engineering and Computer Engineering employer/advisor survey 2008

This survey is part of our process for continuous improvement of the Electrical Engineering and Computer Engineering programs. As an employer/supervisor of our graduates, your feedback is very important to us. Thank you for taking the time to complete this survey. All results will be taken anonymously, and no information will be shared with our alumni.

1. Please choose the description that best describes your position.

Engineering supervisor
HR representative
Graduate school admissions director
Graduate faculty advisor
Other (please specify)

2. Compared with graduates from other institutions, how would you rate the engineering skills of UCR graduates as they enter the work force or graduate programs? (1-Very inferior 5-Very superior)

3. We design our curriculum to enable our graduates to succeed in the following areas. Please evaluate the degree to which you believe your UCR alumni are succeeding in these areas. We also would appreciate your insights into the importance of these areas.

3a. Professional visibility: contributions to your organization and the engineer's profession through publishing and/or patents [1-5 rating for success] [1-5 rating for importance]

3b. Professional achievement, as exemplified in awards or honors from the company or profession patents [1-5 rating for success] [1-5 rating for importance]

3c. Professional responsibility, as exemplified by involvement in professional associations, organizing conferences, or serving as a peer reviewer patents [1-5 rating for success] [1-5 rating for importance]

3d. Professional leadership, as exemplified by promotions, raises, and/or positions of authority and leadership on projects patents [1-5 rating for success] [1-5 rating for importance]

3e. Entrepreneurial activity, as exemplified by initiating new projects within the organization or new ventures of his/her own patents [1-5 rating for success] [1-5 rating for importance].

3f. International activity, as exemplified by successful involvement in international collaborations or projects patents [1-5 rating for success] [1-5 rating for importance]

3g. Community engagement, as exemplified by mentoring, volunteering, or other service patents [1-5 rating for success] [1-5 rating for importance].

4. Are there attributes not listed above that you think should be included?

5. Do you have other comments about your specific UCR graduates, UCR in general, or the attributes that new engineering graduates should have early in their post-bachelor's career? You can enter short answers below or, if you prefer, arrange for a discussion with our faculty and staff by sending an email to the Chair of the Assessment and Accreditation Committee, Prof. Amit K. Roy-Chowdhury, amitrc@ee.ucr.edu.

Thank you for helping us by completing this survey.

Appendix H: Instructions for EE175A/B in 2007-2008 and Three Sample Reports

To illustrate that we have implemented an ongoing process for improvement, we attach here the instructions and student sample reports of EE175A/B as implemented in 2007-2008. The equivalent documents for this were provided in Appendix A of our response to the draft ABET findings.

EE-175: Senior Design Project

Winter and Spring 2008

Class

Lecture:	175A: Mondays 11:10 a.m. - 12:00 a.m. SPTH 2200 Lecture room for 175B TBA
Lab:	M(PL) T(SK) W(YH) Th(MO): 4:10-5pm ENGR2 125

Instructors:

Professor Yingbo Hua	yhua@ee.ucr.edu	EBU-II 432	827-2853
Professor Sakhrat Khizroev	khizroev@ee.ucr.edu	EBU-II 414	827-5816
Professor Ping Liang	liang@ee.ucr.edu	EBU-II 323	827-2261
Professor Mihri Ozkan	mihri@ee.ucr.edu	EBU-II 436	827-2900

Prerequisites

Senior standing in Electrical Engineering.

Objectives

The Senior Design Project is the culmination of course work in the bachelor's degree program in electrical engineering or computer engineering. In this comprehensive two-quarter course, students are expected to apply the concepts and theories of electrical engineering or computer engineering to an engineering project. Detailed written reports, working demonstration, poster and oral presentations are required.

Credits and Hours

Eight quarter units of engineering design credit will be granted for the completed project and other required components listed here. It is expected that approximately twelve hours of laboratory (or field) work will be required weekly for satisfactory completion of the project. The design value of these units has been accounted for in the total number of required science and design units necessary for graduation.

Weekly Class Meetings

The entire class of EE 175A and EE 175B will meet once each week for one hour. These meetings are intended to provide instruction in topics common to all design projects (engineering economics, ethics, etc.). In addition, it is expected that each project team meet with their faculty supervisor on a weekly basis to report and discuss the progress of the project. They may include brief presentations by each team, aimed at improving technical presentation skills. The lectures and weekly meetings are mandatory and are for your benefit (10% of grade).

Project Participants

Projects will be completed in small teams with shared responsibility. If the team option is elected, each student will be held responsible for a distinct component of the total team effort. Team projects will be sufficiently more complex than individual projects so as to allow for an appropriate workload for all team members.

Project Elements

The senior design projects will include proposal and report writing, experiment design, hardware and software design, test plan and test, broad impact and ethical issues, among other things. Remember that this is a design course and students must define a ***design*** project, not a research, nor an evaluation or fabrication project. It is a balanced approach to encompass many of the elements stated above.

Each design project must include the following components:

1. **A Clear Technical Design Objective and the Project Contract** (Contract due 1/14/08): Each group must identify a design project on or before 1/14/08, and should have good estimated answers to the following questions and obtain the endorsement of the section professor:
   - Is the objective achievable within two quarters?
   - Does the group have the expertise to complete the design, prototype, and testing?
   - Does the group have access to the financing for the prototype?
   - Does the group have access to the required test equipment?
   - Is this a design problem (not research, nor fabrication)?
   - Is the project significant enough to be worthy of eight credits (12 hours/week/person)?

2. **Experiment Design and Feasibility Study:** Design and carry out experiments to evaluate the feasibility of project ideas, alternatives, trade-offs and realistic engineering constraints. Analyze the experimental results to prove the feasibility of your project idea and select the best solution to be further developed in the design project.

3. **A Detailed Design Specification** (Due in week 7 in the Winter quarter, 10% of final grade): Describes the functions and quantitatively measurable design objectives, design methods, hardware and software architecture and interfaces, user interface, realistic constraints in terms of time, cost, safety, reliability, social impact, ethics, etc. It must also list and consider the industry standards related to your project, including hardware, protocols, software and tools (e.g., 802.11, RS232, USB, PCI, 3G, API, device drivers, VHDL).

4. **Test Plan** (Due in week 9 of the Winter quarter, 5% of final grade)**:** A detailed description of your design of experiments to test and measure whether the final product and each of its components meet the design specifications, and, if not, to test and measure the errors and deviations from specifications.

5. **Global, Economic, Environmental And Societal Impact** (Due 2/11/08, 2.5% of final grade): Each student must write an essay (800 or more words) providing an analysis of the potential global, economic, societal, and environmental impact of the project. You do not need to address every aspect, just focus on a couple of aspects that are related to your project. For example, if your project is made into a product, how will it improve quality of life, affect the environment, enhance entertainment, education, globalization etc.? Are there any ethical or political debates, laws and regulations that are related to your project?

6. **Contemporary Engineering Issues** (Due 2/25/08, 2.5% of final grade) Write an essay (800 or more words) on the contemporary engineering issues related to the project. Potential contemporary engineering issues related to your project are new technologies, new industry standards, new design methods, new materials, new trends in manufacturing, etc.

7. **Detailed Quantitative Design and Prototype** (To be completed before week 8 of the Spring Quarter) Each component of the selected solution and the overall system should be designed and implemented. In most cases, it is necessary to construct a system prototype (or component prototype).

8. **Test Report** (Due week 10 of the Spring quarter, 5% of final grade): Carry out the Test Plan you developed to identify how well your final design meet the specifications under the defined constraints, and present the results in this report.

9. **Poster and Final Presentation** (Due week 10 of the Spring quarter, 15% of final grade): Each group must prepare a poster and a Power Point presentation, and present the final design to faculty and other students.

10. **Working Demo and Final Report** (Due 6/6/2008 before 5pm, 45% of final grade): The final report must include all the required sections and appendices in a template file to be posted on the iLearn website for the course. A working demo of the completed design is critical, it is a convincing evidence that you design is completed and works. The demo should show whether and how design specifications are met.

Grading

In addition to the 9 deliverables listed above, each project will also be graded on the following:

1. **Laboratory Notebook, Weekly Reports and Lecture Attendance:** Each student team need to maintain a laboratory notebook for the duration of their projects and submit written weekly reports. This notebook and reports will be inspected at weekly meetings and graded for content. Attendance of the lectures is mandatory. Everyone must sign in at each lecture. (This portion accounts for 10% of grade)
2. **Ethics Exam**: 5% of the final grade

Grading will be determined by all of the section professors conferring on each project and student. Please note that grades are assigned to an individual, not to a project.

Project Topics

Projects will be carried out in four different sections corresponding to the main electrical engineering areas taught at UCR. Each section will have a "section professor" (i.e., faculty supervisor). Possible project topics are obtained from or approved by the section professor. In addition, joint projects with other departments may be arranged.

Electrical Engineering Area	Section Professor	Topics
Nano-Materials, Devices and Circuits (NMDC)	Sakhrat Khizroev	1) Nuclear Magnetic Resonance Spectroscopy 2) Magnetic Levitation (train and others) 3) Environmentally friendly magnetic/ electromagnetic car
Nano-Materials, Devices and Circuits (NMDC)	Mihri Ozkan	**Solarium Environment and Solar Cell Development—**Students will build an environmental test chamber (i.e., Solarium) that is capable of simulated sunlight at different intensities. The students will then fabricate organic solar cells and subsequently characterize these designs in the environmental test chamber.
Computer Engineering (CE), Controls and Robotics (CR)	Ping Liang	Embedded systems and wireless devices: 1. wireless controllers for video games, sensor-based smart toys, networked sensors 2. sumo robots, micromouse, GPS based autonomous vehicles
Communications and Signal Processing (CSP), Controls and Robotics (CR),	Yingbo Hua	1. spectral analysis of bio-logical signals 2. statistical analysis of internet traffic 3. acoustic sensor array processing for smart robots

Steps in Selecting a Project

Upon reviewing the topic areas, students should take the following steps to select a project, and sign the corresponding senior design contract (next page).

Step - 0: Prepare a brief academic resume, which describes your specific technical strengths and general background in less than two pages. It is very important that you make a case for yourself as why you should be doing a specific project. This step is more or less like applying for a job, and therefore this resume is the first draft of your future resume that opens a door for you. Then follow one of the following Steps 1A to 1C, depending on your situation.

Step - 1A: Meet and talk to your section professor, and find out if the professor offers an project that interests you and he/she considers you qualified to do the project. Or,

Step - 1B: If you have an industrial project in mind that meets the requirements stated above, then you still need to talk to your EE175 section professor. This professor must approve and supervise the project.

Or,

Step - 1C: If you have your own project, you must lobby for that idea with your section professor. This approach requires additional effort, but is doable if it is planned in advance.

Step - 2: Identify one or possibly two of your classmates who have similar interests and want to work with you on the same project and have gone through the same steps as you did. Discuss the project among team members and achieve a consistent project idea.

Step - 3: Make a brief written proposal to the section professor that includes your resume, your classmate(s) resume(s) if applicable, the title of the project, and a brief description. Also have one or more projects in this proposal as your second or third choices. Please note that every effort will be made to match you with your best choices, although in certain instances changes may be required. In that case you will be notified promptly.

Step – 4: Once the projects are verbally approved by the section professors, each student team is to fill out a project contract available on the class web site. Be sure to fill out every section, sign it, and turn it in to your section professor.

Weekly Lecture Class Schedule

Date	Event	Lecturer	Lecture Content
1/7	Week 01	All	Introduction, course outline, preliminary issues, requirements and expectations
1/14	Week 02	YH	Design methodologies and approaches; block diagrams, analyses of solutions, evaluation of feasibility.
1/21	Week 03		Holiday, no class
1/28	Week 04	SK	Experiment design, developing a test plan, collecting data, and evaluation. Design constraints, industry standards
2/4	Week 05	MO	Technical writing
2/11	Week 06	SK	Introduction to the design process, specification process, laboratory notebooks, library techniques, literature and information search
2/18	Week 07		Holiday, no class
2/25	Week 08	DG	Project management: organization, teamwork, scheduling, budgeting, etc.
3/3	Week 09	MO	How to give oral presentations, preparing for the design review
TBD	Week 10	All	Design Review. Two parallel sessions
3/31	Week 01	DG	Printed circuit board design, layout, and fabrication
4/7	Week 02	MO	Contemporary engineering issues, societal, environmental and cultural impact, international engineering projects
4/14	Week 03	SK	Engineering ethics (exam required)
4/21	Week 04	PL	Intellectual properties
4/28	Week 05	YH	Career choices and strategies, how to write resumes
5/5	Week 06	PL	Engineering economics, marketing engineering products
5/12	Week 07	YH	Final testing requirements, test report, preparation for the final presentation
5/19	Week 08	PL	Entrepreneurial, venture capital and start-ups
5/26	Week 09		Holiday, no class
TBD	Week 10	All	Final Presentations. Demo required. Two parallel sessions

Lecturer code: **DG** – Dan Giles, **YH** – Yingbo Hua, **SK** - Sakhrat Khizroev, **PL** – Ping Liang, **MO** – Mihri Ozkan

Three project reports (getting A+, B and C grades) are attached in Appendix J.

Appendix I: Ability to Function in Multi-disciplinary Teams in EE/CSE120A/B

The students in this course belong to three major: Electrical Engineering, Computer Engineering and Computer Science students. The Lab teams were built such that we try to put students with different majors into a single multi-disciplinary team. For instance, in EE/CS120a Fall 2007, we had 34 students. Among them, there were 19 EE students and 15 CS /CE students. There were 15 teams, each of them multi-disciplinary. As an example, one team consisted of two students: Don Albert Robison and Carl Zheng Hua. Don is EE student and Carl is CE student. Due to the even distribution of EE and CS/CE students, it was not difficult to create multi-disciplinary teams for this course. Such a plan will be followed in subsequent offerings of this class.

Appendix J: Sample Student Projects from EE 175

EE175AB Final Report

TrackLiveGPS.com

EE 175AB Final Report

Department of Electrical Engineering, UC Riverside

Project Team Member(s)	Joseph Silva (SID: 860-202-593)
Date Submitted	6/9/2008
Section Professor	Dr. Yingbo Hua
Revision	3.0
URL of Project Wiki/Webpage	http://www.jtechgps.com , http://www.tracklivegps.com

Summary

This report presents the TrackLiveGPS design project. The project entails the creation of an AVL (Automatic Vehicle Location) device that is used to track vehicles in real time via a web interface. This report will describe the design theory, processes, implementation and results of the TrackLiveGPS device.

Revisions

Version	Description of Version	Author(s)	Date Completed	Approval
3.0	Final Revision. Final Proof.	Joseph Silva	6/8/2008	
2.0	Second Proof. Addition of Testing Results	Joseph Silva	6/5/08	
1.0	First Proof. Completion Of Most Sections	Joseph Silva	5/30/08	

Table of Contents

1 * Executive Summary

The TrackLiveGPS® design project entails the development of an AVL (Automatic Vehicle Location) device. The final product will be a device that can be embedded inside a vehicle, and tracked live time via a web interface. The web interface will allow users to view all their registered devices, view live tracking information such as position, speed, heading, last update time, etc… and also allow them to replay all past tracking information.

The TrackLiveGPS device will use GPS as the source for all required position, speed and heading information. A cellular GSM & GPRS network protocol should be used as a communication link to relay positional information back to a VMC (Vehicle Management Center). The utilization of an IP/UDP protocol will be used to relay the positional information back to the VMC that will store the positional information.

The VMC (Vehicle Management Center) will consist of three servers. Server 1, will be used as the web server to host the TrackLiveGPS.com website. Server 2 will be used to handle all incoming UDP packets from the TrackLiveGPS Devices and store the positional information into a MySQL database. Server 3 will host the MySQL database that will be used to store all positional information.

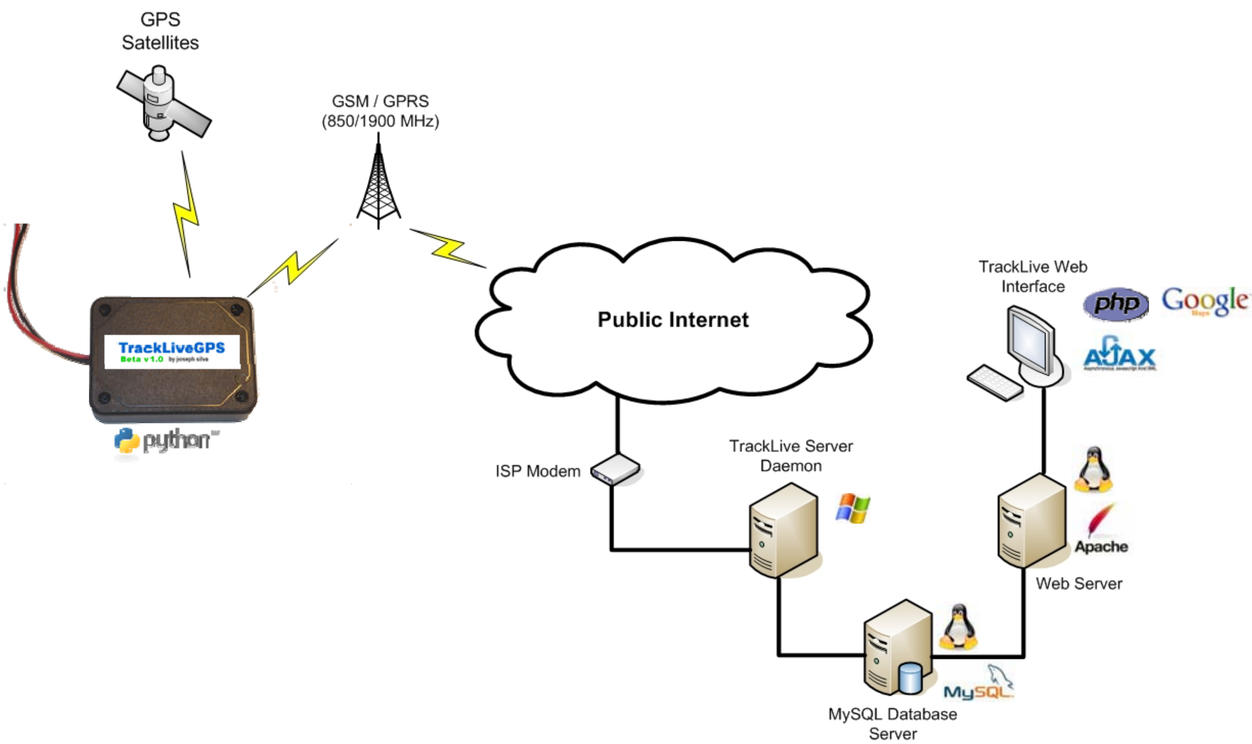
The TrackLiveGPS.com web interface should have options to view "live" tracking information as well as "past" tracking information. The user should be able to select the vehicles he or she wants to track, and view live position information (up to 20 second delay) as well as replay all past location trips.

The AVL proved to be successful and passed all tests successfully. The system has an average reliability of 95% uptime. This is based on a valid GPS fix and communication on the GSM network. Users are able to view there vehicle in real time via the TrackLiveGPS.com web interface. They can view their vehicles position with up to ten second delays. They also have the option to view past tracking information. They can select the day they want to replay on a simple to used calendar. Once the user picks the day of interest they can replay it. They have the option to pause, rewind, or fast-forward at any point.

2 * Introduction

This space may be used to provide an introduction for the design and ties to other project materials.

2.1 * Design Objectives and System Overview



The TrackLiveGPS project will entail the development of an AVL (Automatic Vehicle Location) device. The primary objective of the design project is to develop a device that can be embedded into a vehicle and tracked in "live" time via a web interface. The users will also be able to view and replay past tracking information via the web interface. The system uses GPS to obtain position information. The position information is then relayed back to the VMC (Vehicle Management Center) where it is stored into a database for displaying "live" and "past" tracking information via the TrackLiveGPS.com web interface.

Although there are numerous applications for AVL devices, security and fleet tracking will be the primary application for this project. The device has a small form factor, so it can be easily embedded into the dash of a vehicle and discretely used for tracking.

Technical Design Objectives:

- Product Packaging
  - 3 in x 2 in x 1.5 in (LxWxH)
  - Weight < 8oz (0.5 lbs)
- Reliability
  - Must track 95% of the time (depends on GPS reception and/or GSM reception)

- Data Transfer
  - Less than 10MB per month
  - Updates position information every ten seconds
- Power Consumption
  - Less than 2W
  - About 250 mAh in active GPS and GPRS connection
- GPS Accuracy
  - < 2.5m (~8 feet)
- Easy To Use Web Interface
  - Easy to use web based live tracking
  - Ability to view and replay tracking data from previous days.

2.2 * Development Environment and Tools

This Project entailed the usage of many IDE's and software platforms. The primary hardware and software tools used in the development of this project have been listed in the table below. To program the TrackLiveGPS device the Telit Python IDE was used. For programming the server daemon of the VMC, Visual Basics was used. For hosting the website, a Linux/Fedora Core 8 web server was used. The positional information was stored in a MySQL database. Dreamweaver was used to develop the web interface. For a visual reference to software tools used, please refer to the system diagram in section 2.1.

The EVK2 Development Board is used for the programming of the Telit GM862. A DMM and DC power supply is utilized for testing and development purposes. A reflow skilled and hot air rework station is used for mounting all chip components on PCB's.

Hardware Tools	Software Tools
Telit EVK2 Development Board	Visual Studio 2005
DMM	Telit Python 1.5
DC Power Supply	Macromedia Dreamweaver
Reflow Skillet	Macromedia Fireworks
Hot Air Rework Station	Matlab
Soldering Iron	Eagle Cad
Solder Paste	Fedora Core 8
Digital Calipers	Windows Server 2003
	Windows XP Pro
	MySQL
	Apache
	PHP
	Javascript
	Google Maps API

2.3 Related Documents and Supporting Materials

Telit GM862 AT (v.4) Commands Reference (http://www.gm862.com)

2.4 * Definitions and Acronyms

The Acronyms defined below will or have been used throughout this document:

GPS – Global Positioning System

VMC – Vehicle Management Center

GSM – Global System for Mobile communications

GPRS – General Packet Radio Service

AVL – Automatic Vehicle Location Device

AJAX – Asynchronous Java and XML

XML – Extensible Markup Language

PHP – Hypertext Preprocessor

Apache – Apache HTTP Web Server Software

3 * Design Considerations

This section describes issues that need to be addressed or resolved prior to or while completing the design as well as issues that may influence the design process.

3.1 * Assumptions

For the TrackLiveGPS we are making assumptions about the operating environment of the module. We assume that the power source used to power the device will be available in the operating ranges of 3.7-35V DC, and be able to supply up 2 A. Another assumption is that power will be continuous, as any drops in power during tracking will cause a hardware reset on the device.

The assumption is made that the device will be in a low noise situation and have clear communication with cellular towers and a clear reception of GPS signals. If the device is not in range of a cellular communication tower the device will not be able to relay position information back to the VMC. Also, if it is not possible to receive a reliable GPS signal, positional information will not be able to be relayed back to the VMC.

The system also assumes that the VMC will always have a stable connection to the internet. If at any time the VMC goes down due to connectivity issues, all tracking data sent during the outage will be lost. In order for the system to work successfully the VMC, GSM, and GPS must all work at the same time.

To view the tracking via the web interface, it makes the assumption that users will have a broadband connection, Internet Explorer 6.0+, Fire Fox, or Safari web browser. The user should also have a operating system of windows XP or greater. The system should have a PIII - 500MHz processor or greater with, minimum 128MB ram.

3.2 * Realistic Constraints

Hardware Constraints:

Operating Voltages: 3.7-35V DC

Supply Current: 0-2A DC

Operating Temperatures: 0-180 F

Communication: Clear reception of GPS signals, within range of GSM, GPRS availability

No floating point arithmetic on the Telit GM862 module.

VMC Constraints:

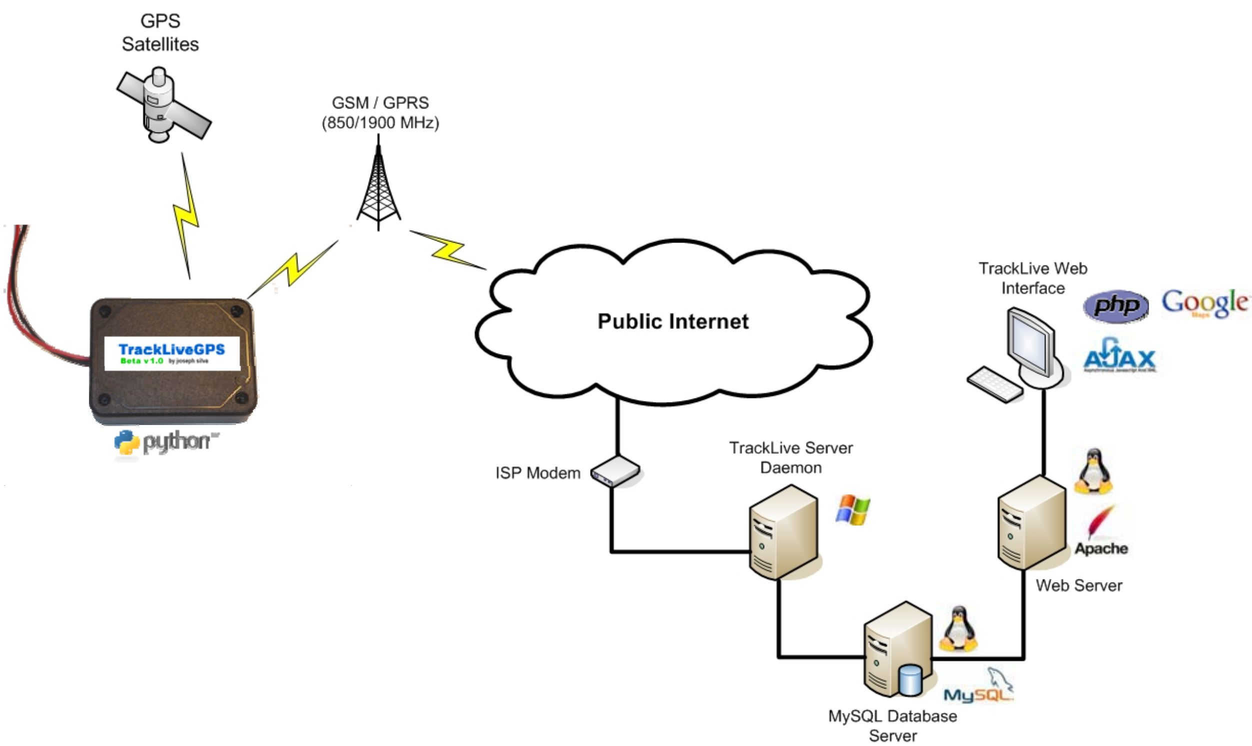
Uninterrupted Power Supply & UPS Backup

Reliable connection to internet.

Well ventilated and cooled environment for VMC hardware.

3.3 * System Environment and External Interfaces

The System and environment can be best described by the model below. The TrackLiveGPS module has a python interpreter built-in. It uses GPS and GSM as a source for positional and communication protocol. The VMC is connected through the IP through an ISP Modem. The positional information is sent from the TrackLiveGPS device via UDP and parsed by the TrackLive Server Daemon. The information is then stored in a MySQL database. The location is then finally viewed by the user on the TrackLiveGPS.com website. By using Google Maps API, along with PHP and AJAX, the user can see “live” and “past” position information.



3.4 * Industry Standards

Global Positioning System (**GPS**) is the only fully functional Global Navigation Satellite System (GNSS). Utilizing a constellation of at least 24 Medium Earth Orbit satellites that transmit precise microwave signals, the system enables a GPS receiver to determine its location, speed, direction, and time.

Used Frequencies For The TrackLiveGPS device:

- L1 (1575.42 MHz): Mix of Navigation Message, coarse-acquisition (C/A) code and encrypted precision P(Y) code, plus the new L1C on future Block III satellites.

User Datagram Protocol (UDP) - is one of the core protocols of the Internet protocol suite. Using UDP, programs on networked computers can send short messages sometimes known as *datagram’s* (using

Datagram Sockets) to one another. UDP is sometimes called the **Universal Datagram Protocol**. The protocol was designed by David P. Reed in 1980.

Transmission Control Protocol (**TCP**) is one of the core protocols of the Internet protocol suite. TCP provides reliable, in-order delivery of a stream of bytes, making it suitable for applications like file transfer and e-mail. It is so important in the Internet protocol suite that sometimes the entire suite is referred to as "the TCP/IP protocol suite." TCP is the transport protocol that manages the individual conversations between web servers and web clients. TCP divides the HTTP messages into smaller pieces, called segments, to be sent to the destination client. It is also responsible for controlling the size and rate at which messages are exchanged between the server and the client.

General Packet Radio Service (GPRS) is a packet oriented Mobile Data Service available to users of Global System for Mobile Communications (GSM) and IS-136 mobile phones. It provides data rates from 56 up to 114 kbit/s. GPRS can be used for services such as Wireless Application Protocol (WAP) access, Short Message Service (SMS), Multimedia Messaging Service (MMS), and for Internet communication services such as email and World Wide Web access. GPRS data transfer is typically charged per megabyte of traffic transferred, while data communication via traditional circuit switching is billed per minute of connection time, independent of whether the user actually is utilizing the capacity or is in an idle state. GPRS is a best-effort packet switched service, as opposed to circuit switching, where a certain Quality of Service (QoS) is guaranteed during the connection for non-mobile users.

Global System for Mobile communications (**GSM**: originally from *Groupe Spécial Mobile*) is the most popular standard for mobile phones in the world. Its promoter, the GSM Association, estimates that 82% of the global mobile market uses the standard. GSM is used by over 3 billion people across more than 212 countries and territories. Its ubiquity makes international roaming very common between mobile phone operators, enabling subscribers to use their phones in many parts of the world. GSM differs from its predecessors in that both signaling and speech channels are digital, and thus is considered a *second generation* (2G) mobile phone system. This has also meant that data communication was easy to build into the system.

Hypertext Transfer Protocol (**HTTP**) is a communications protocol for the transfer of information on the intranet and the World Wide Web. Its original purpose was to provide a way to publish and retrieve hypertext pages over the Internet.

3.5 * Budget and Cost Analysis

Present your budget and/or cost analysis.

TrackLiveGPS Module (Per Module)

Item	QTY	Total Cost
Telit GM862 Module	1	$140.00
Plastic Enclosure	1	$2.50
PCB (Silkscreen & Soldermask)	1	$12.00
GPS Antenna	1	$12.00
GSM Antenna	1	$9.00
Analog Components(R,C,IC)	1	$8.00
Power Cables, Connectors	1	$4.00
SIM Card	1	$6.00
TOTAL COST		$193.50

VMC (Vehicle Management Center)

Item	QTY	Total Cost
AMD Quad Core, 8GB,Raid 1 300GB, Rackmount Servers w/ Rack, Monitor, Switch, KVM, UPS, RM Keyboard. All hardware included	3	$2800.00
Windows Server 2003 Trial	1	$0.00
Fedora Core 8	1	$0.00
Visual Studio 2005, Student Ed	1	$130.00
TOTAL COST		$2930.00

GSM/GPRS Service (Per Module)

Item	QTY	Total Cost
Sim Card	3	$6.00
5MB Data Plan (9 months)	3	$8.00/Month
TOTAL COST		$18.00 + $24/month

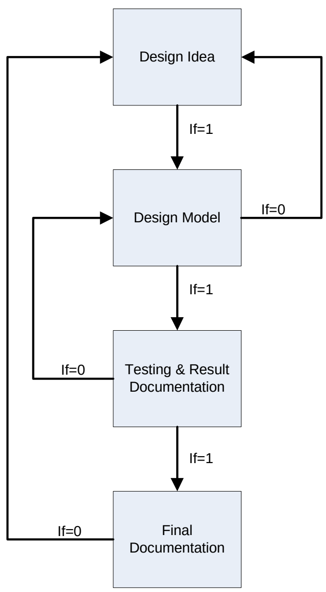
3.6 * Safety

Discuss safety considerations and specify safety objectives

Since the TrackLiveGPS devices acts as a passive system that is embedded inside vehicles it does not pose a large safety threat (In terms of control). The only safety hazards stem from the voltage regulation circuitry that is built into the TrackLiveGPS hardware. Since the power systems of vehicles are susceptible to noise, power spikes and temperature, the TrackLiveGPS device must be able to handle all possible operating conditions to ensure safe operation.

In the design of the voltage regulation voltage spikes, current regulation, polarity inversion, and temperature conditions were all taken account for. It should also be noted that all Analog Components (Resistors, Capacitors, etc…) should be selected with power voltage and power ratings. All capacitors will be selected with a voltage rating of two times the predicted voltage range.

3.7 * Documentation



The design process entails many steps, but can be summarized by the flow chart to the left.

Design Notes and Engineering Change Notices Protocol: During the first two phases of the design process, if any changes or modifications are made they should be documented in the design notes of the technical documentation.

Version Update: if during the final documentation, a change in the model is needed, then a version update should be made, ie. (1.1 to 1.2). This should happen if the Design model has to be modified after the testing and results process.

Version Control: if during the final documentation a major change has to be made to the model , then the version should be changed, ie (1.0-2.0).

Technical Documentation: all changes made at any point of the design process should be relayed in the technical documentation.

User Documentation: During the final documentation phase, all user documentation should be generated. If at any point the module has a version change, then the user documentation should be updated accordingly.

3.8 * Global, Economic, Environmental and Societal Impact

The TrackLive device can and will impact the world and society on many levels. It will have the largest impact on economic and societal aspects of the world. By using the TrackLive device for security purposes, it has potential for a great positive economic impact on the world. When the TrackLive device is used to track a vehicle for reasons other than security, there can be many positive and potentially negative societal and ethical impacts on society.

When the TrackLive device is used for security purposes, it has the potential of positively impacting the economic lives of everyone. As time goes on, more and more vehicles are stolen each day. According to a 2006 report by the US FBI Uniform Crime Reports, 1,192,809 vehicles were reported stolen in the United States. Stolen vehicles affect not only the owner of the car, but everyone who buys auto insurance. If more and more vehicles are stolen and not recovered, the insurance companies will have to raise premiums in order to cover their losses, producing a negative economic impact on consumers in US and the world. If consumers have to spend more money on insuring vehicles then they will have less money to spend in other parts of the economy. Since insurance is mandatory in the all 50 states of the US, higher premiums can have a substantial negative impact on consumers and the economy.

By offering a security system that can locate a vehicle on demand (such as the TrackLive device) will allow authorities to recover many (if not most) of vehicles that are stolen. Having the ability to track and recover stolen vehicles, will substantially reduce the losses occurred by the insurance companies. By reducing the losses of insurance companies, the insurance companies can then reduce the insurance premiums. By lowering insurance premiums consumers will have more money to spend in other parts of the economy. The overall affect will produce a positive impact on the entire global economy.

The initial cost of the TrackLive device will be the largest expense for most consumers, since most AVL(Automatic Vehicle Locating) devices can range in price from $200 US - $800 US, and have a monthly service fee of ($5.00 US - $60.00 US). The monthly service fees can also be a fairly large cost for consumers. Most insurance companies do offer discounts on insurance premiums for vehicles with AVL devices, so the consumer can offset the monthly service fees for the tracking device. Once AVL's become standard and most vehicles are equipped with them, the initial cost and monthly services will be driven down for consumers. It will take time for the global economy to see a substantial positive economic effect of AVL's. In the long run AVL's have the potential to save consumers millions (or even billions) of dollars, freeing up resources for other parts of the global economy.

The TrackLive device, like other AVL's can have a potentially positive and negative effect socially on the world. Some of the positive effects are the ability to track lost vehicles and persons. It also can serve as a way for parents to track the activity of their children. Employers can also use the device to ensure that employees are using company vehicles for proper use.

On the other hand, the device could be potentially used for bad or unethical reasons. For example, if a stalker became obsessed about someone, they could implant an AVL device on that persons car to track their vehicle. Some people might also see the device as an invasion of privacy, since it is possible track and view all their movements in their vehicle. Also, on ethical debate that could arise, is how and who has access to the tracking data. Can the government use the tracking data? Can the justice system use the tracking data? If the consumer is paying a private company for the service does the government or justice system have the right to view the information? These are just some of the ethical and societal impacts and questions that could arise from the widespread use of AVL's. It is difficult to accurately predict the exact societal impacts of using AVL's, but it is possible to predict the various ethical debates that could arise.

Overall, the TrackLive device and other AVL's on the market can positively impact the economic and societal aspects of the world. It is debatable how to device will ethically impact the societal aspect of the world. At this time, we believe that the benefits of using the TrackLive device and other AVL's on the market far outweigh the negative societal effects of using the device.

3.9 * Contemporary Engineering Issues

In every new design project there are many things to consider. The design engineer will need to look at contemporary engineering issues related to their project. Some of these issues can include new technologies, new communication standards, new materials and new trends in manufacturing. It is a necessity that the engineer researches these issues and adapts these issues into the design.

AVL devices have traditionally been very expensive. This is due to expensive GPS and GPRS modems. Not only was the initial hardware expenses high, the monthly service fees for live tracking was also very expensive. In the past few years GPS hardware has had a dramatic decrease in size and price. GPS modules are now smaller and less expensive than in the past. Along with more inexpensive hardware, there has also been a dramatic decrease in GSM/GRPS communication costs. By offering lower cost hardware and less expensive monthly data fees the TrackLiveGPS design project will offer one of the least expensive AVLs on the market. Along with a relatively low cost AVL, the monthly service plans will be as much as 75% less the competitors.

The GPS modules on today's market are also becoming more sophisticated. They are starting to offer differential GPS. This means that the modules will have some type of corrections (either code or carrier). These differential corrections will lead to more accurate GPS units. The Telit GM862 use WAAS or Wide Area Augmentation System, which offers a free differential correction.

GSM and GPRS services have become less expensive over the past few years, making AVL and telematics more affordable. GPRS has been around for many years and has served the 2G and 2.5G GSM networks. The 2G and 2.5G networks operates in the range of 56 to 114 kbps. 3G which is a new technology will offer speeds from 284 to 2Mbps. The 3G protocol will only operate on the new UMTS standard and will not utilize the GPRS standard.

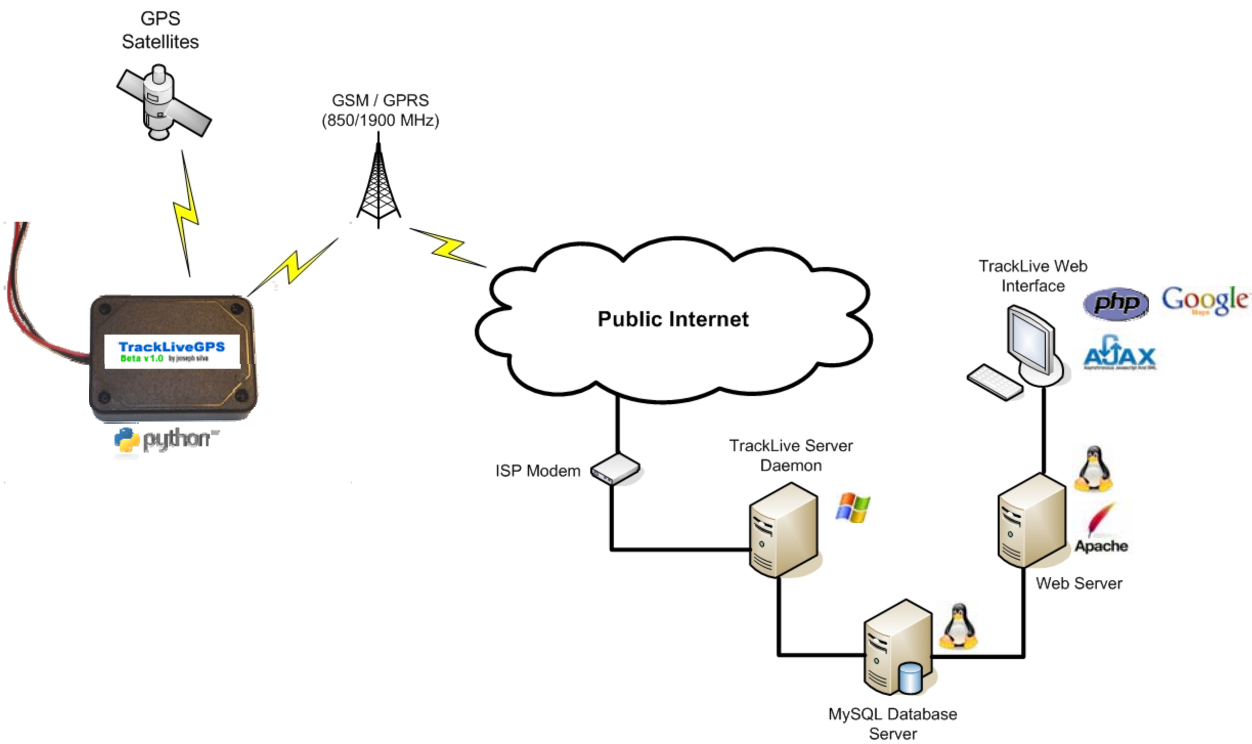
The new general trend of 3G will eventually lead to a shift from GPRS to UMTS standard. This will lead to a shorter lifespan of the TrackLiveGPS device, as it utilizes the 2G GPRS standard. As the 3G technology evolves, less expansion will be made on the GPRS standard, leading to a decrease in availability and eventually reliability of the GPRS networks.

The TrackLiveGPS device will be limited to the availability of current GPS and GSM standards. If in future changes the GPRS protocol is removed or the L1 GPS is updated, the TrackLiveGPS device will be obsolete. The device will be extremely dependent upon the availability of L1 GPS and GPRS standards.

For future work it would be important to try to adapt the current system to the new upcoming technologies in GSM and GPS. Developing a system that can utilize the 3G networks will provide faster connections, increased bandwidth and potentially a more reliable system. With L2 carrier phase corrections becoming more available and less expensive users can expect centimeter level accuracy.

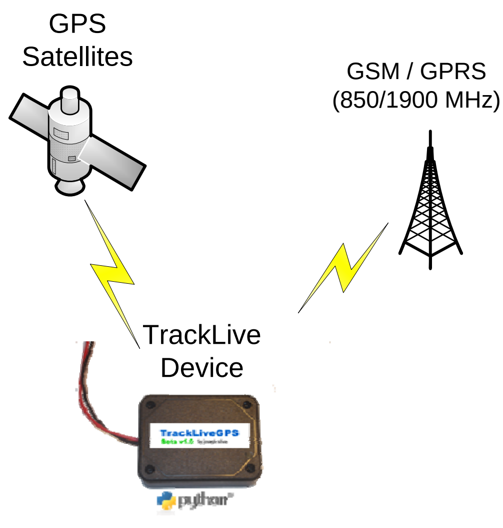
4 * High Level Design

This section describes in further detail elements discussed in the Architecture. Normally this section may be split into separate documents for different areas of the design.



4.1 TrackLiveGPS AVL Device

4.1.1 Conceptual View

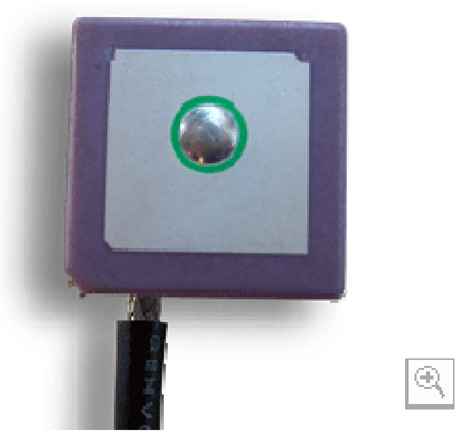


4.1.2 Hardware

AVL Packaged

AVL Cover Off

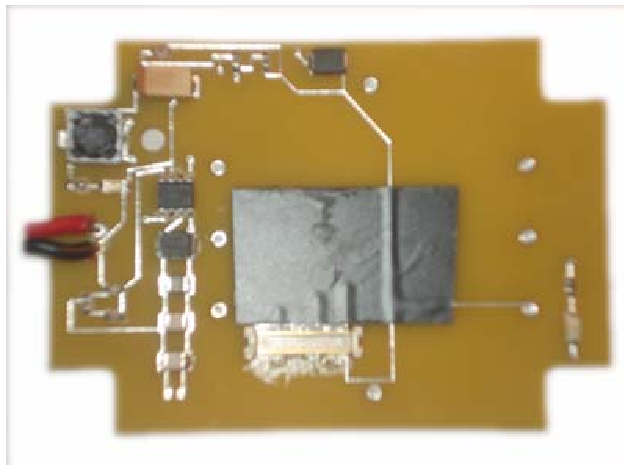
GPS antenna



GSM Antenna

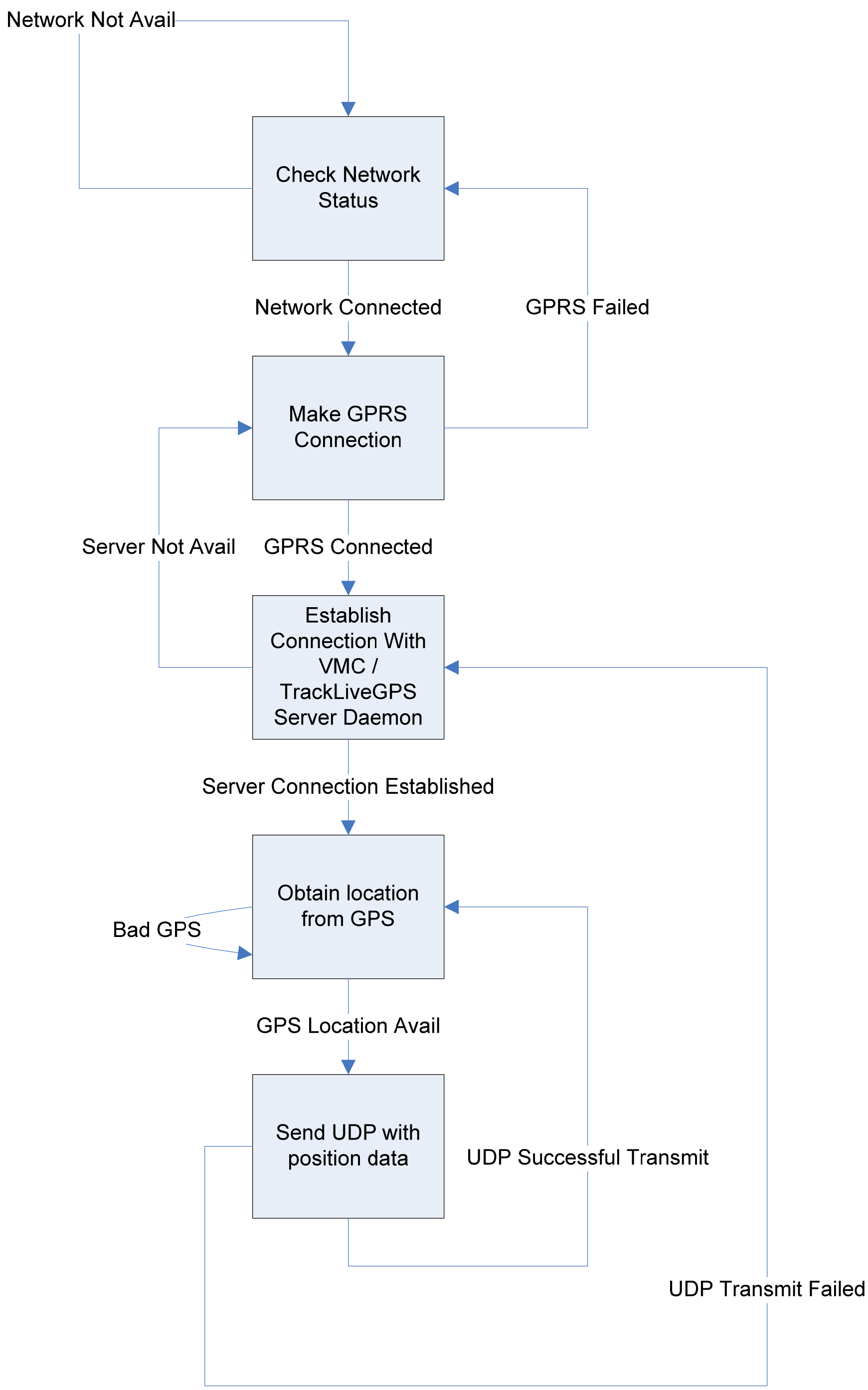
Telit GM862-GPS (Cellular Modem & GPS Module)

TrackLiveGPS PCB



Plastic Enclosure

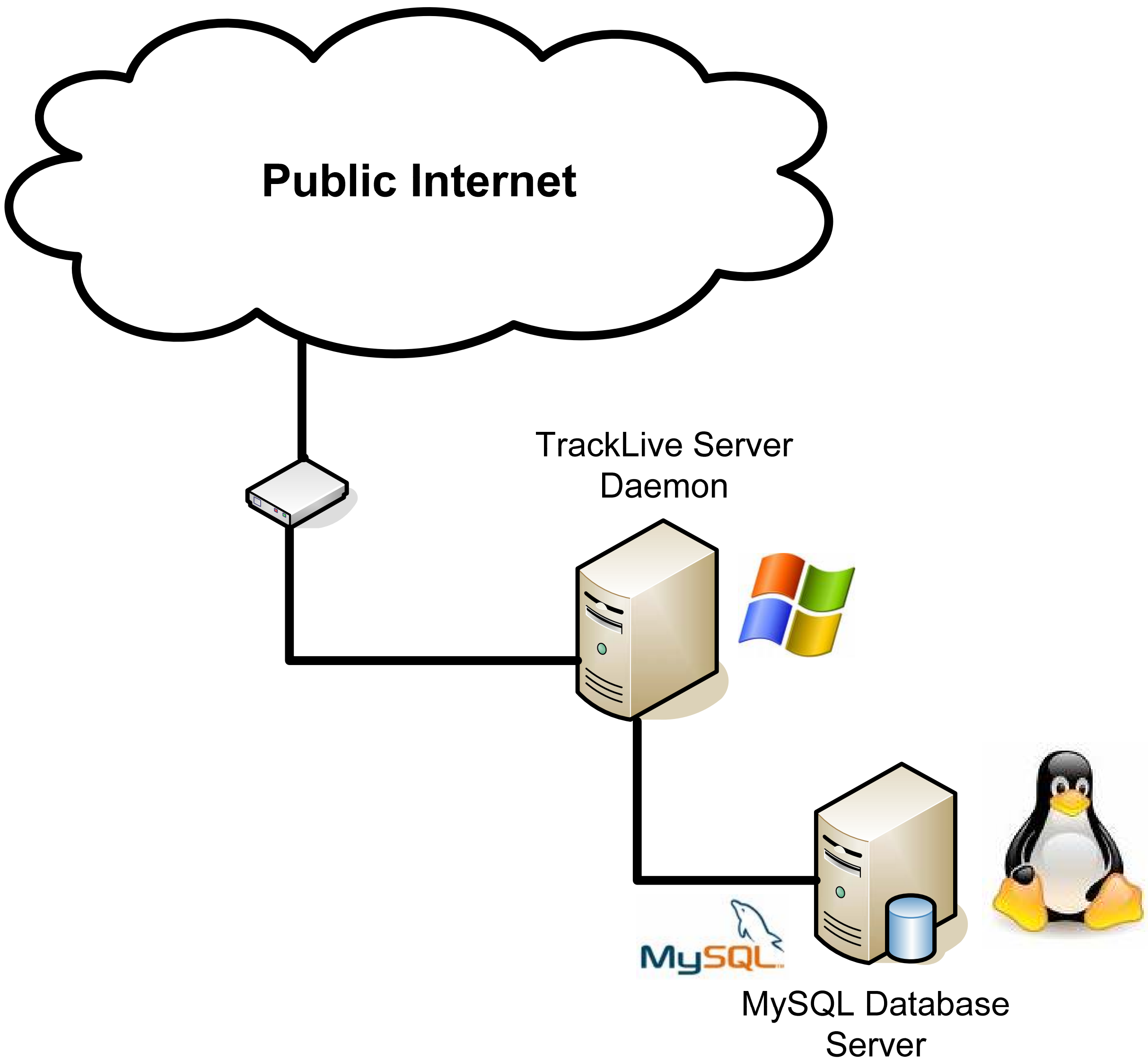
4.1.3 Software



4.2 TrackLiveGPS Server Daemon

4.2.1 Conceptual View

As shown in the diagram below the TrackLiveGPS Server Daemon is connected to the internet. It accepts all incoming UDP packets from the AVL devices. This system operates on a Windows Server 2003 operating system. The Visual Basics application that was written for the project and runs on the Windows Server, acting as the UDP Daemon for the TrackLiveGPS project. The MySQL database operates on a server that is running Fedora Core 8, and runs MySQL 5.0

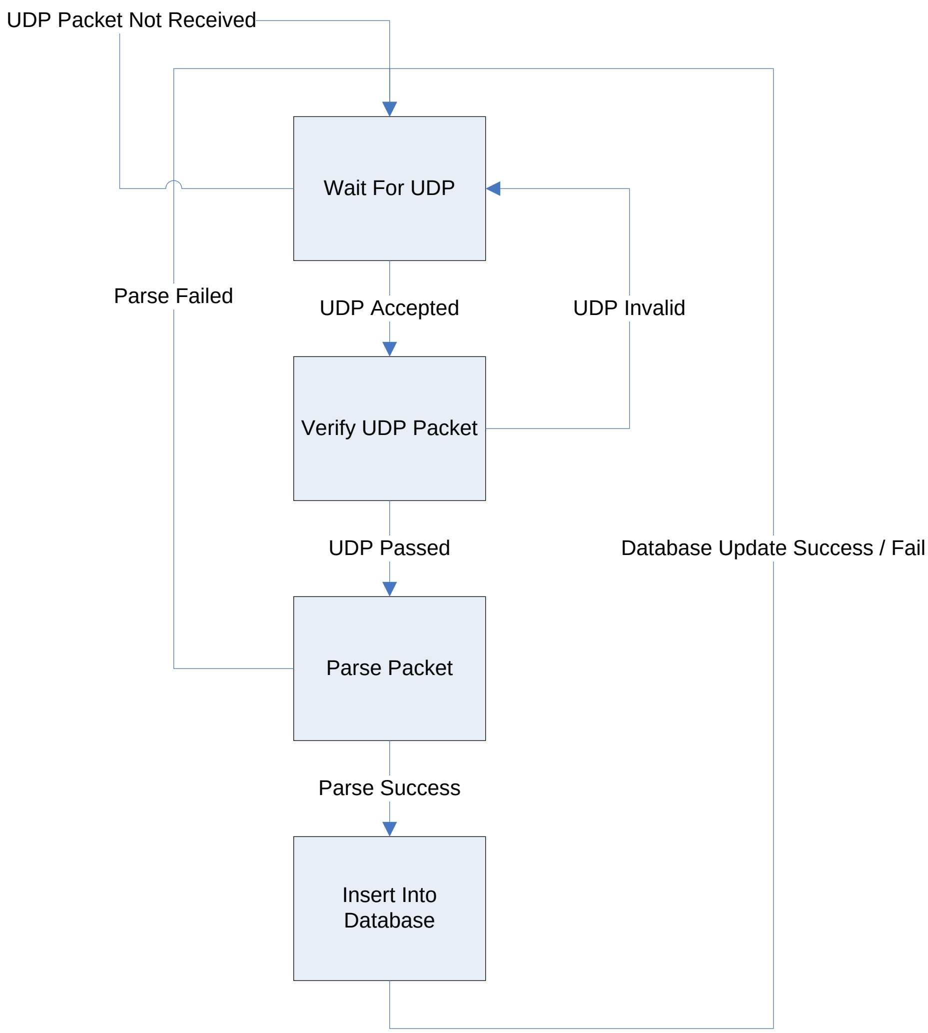


4.2.2 Hardware

The hardware below shows the main components of the Server Daemon. The Actual device depicted in the picture below is the middle rack unit.

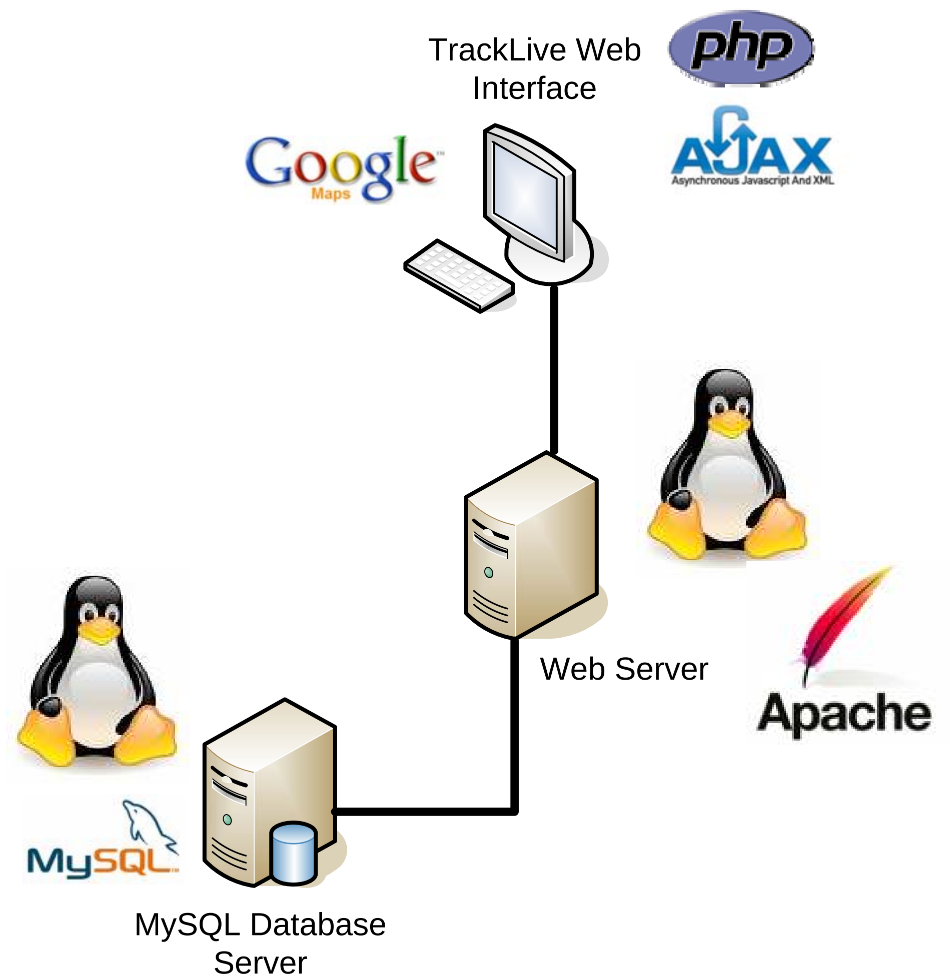
4.2.3 Software

The program flow overviews the Visual Basics software written to manage all incoming UDP packets from the AVL devices. The program flow shows how the system handles the packets and inserts the information into the MySQL database.



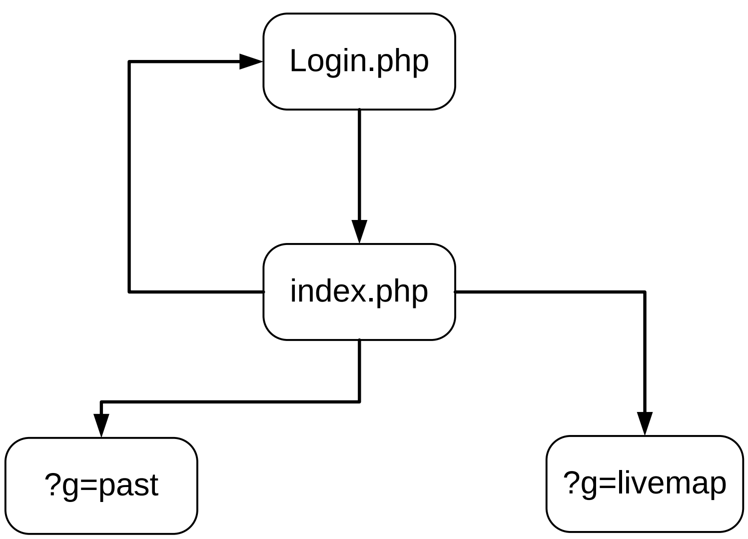
4.3 TrackLiveGPS.com Web Interface

4.3.1 Conceptual View



4.3.2 Hardware

4.3.3 Software

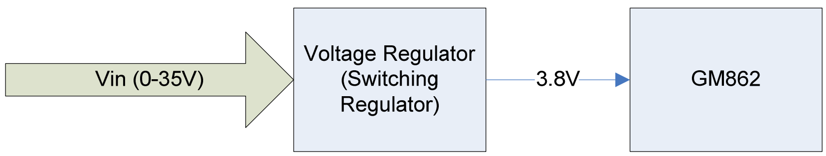


5 * Low Level Design

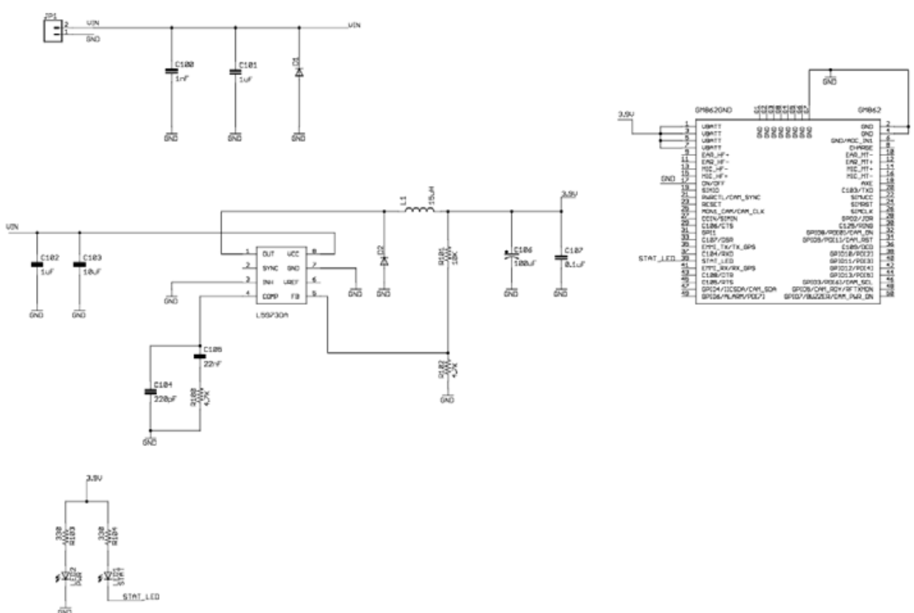
5.1 TrackLiveGPS AVL Device

5.1.1 Hardware Design

The diagram below represents the model for the TrackLiveGPS AVL device. It simply consists of a switching regulator circuitry to regulate the supply voltage for the Telit GM862 to 3.8V. The desired output for the power supply is 3.8V, hence due to the big difference between the input source and the desired output, a linear regulator is not suited and shall not be used. A switching power supply will be preferable because of its better efficiency especially with the 2A peak current load represented by the GM862-GPS.

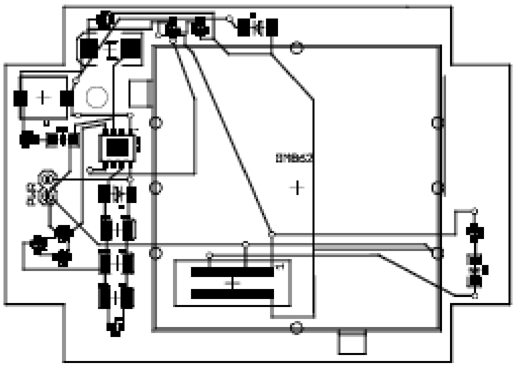


Schematic:

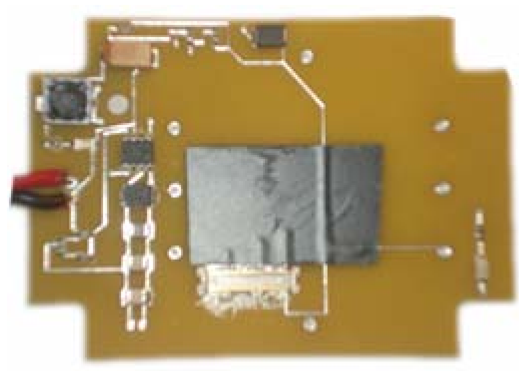


Layout:

Cad Layout



PCB with IC and Analog Componnets

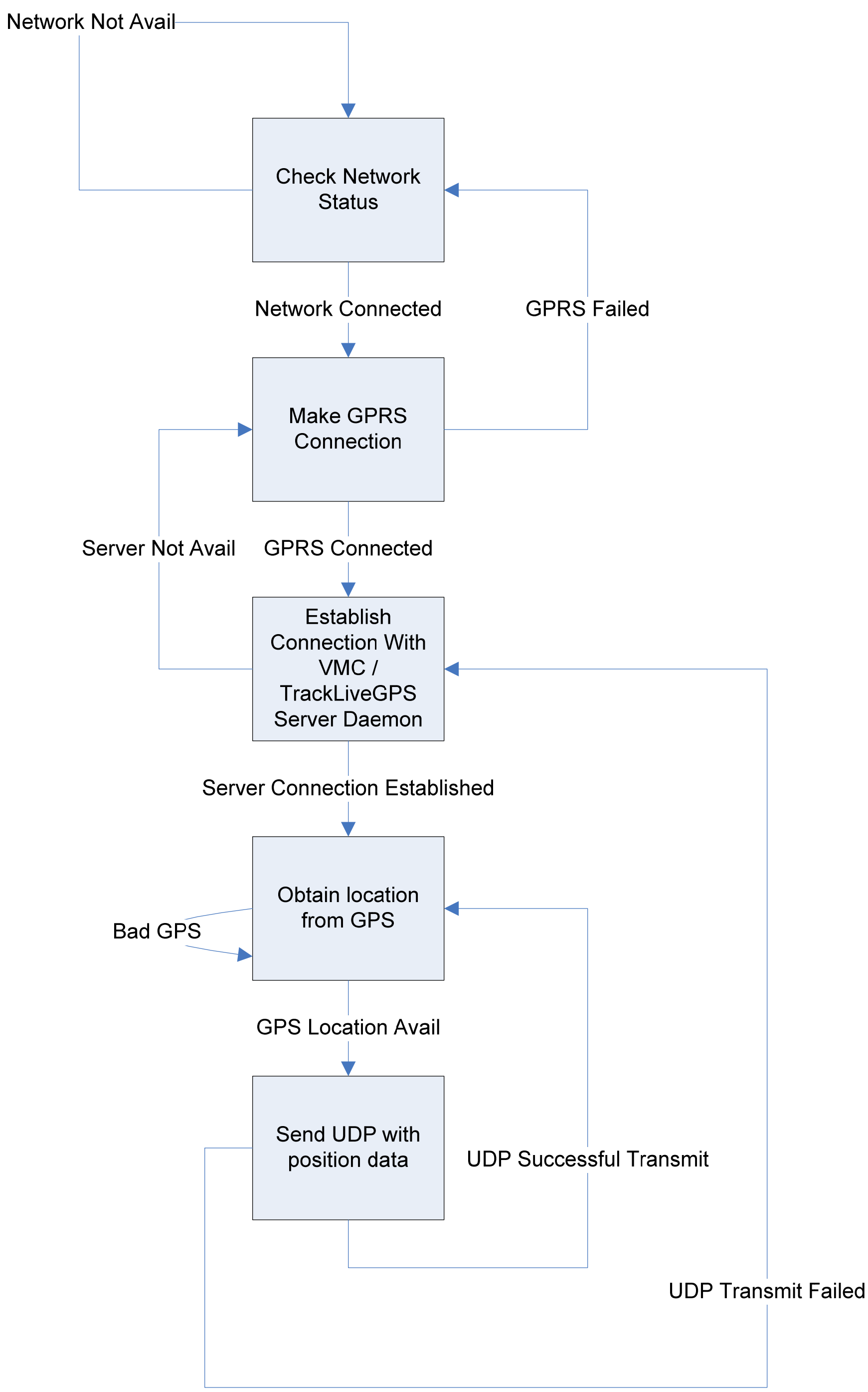


PCB with GM862

AVL (PCB,Enclosure, Antennas)

5.1.2 Software Design

The AVL device is programmed via python language. The GM862 can be programmed in a high level language. This allows for rapid deployment and easy debugging. Also, exception handling and an object oriented language also produces a more reliable end product. The diagram below describes software flow for the AVL device.

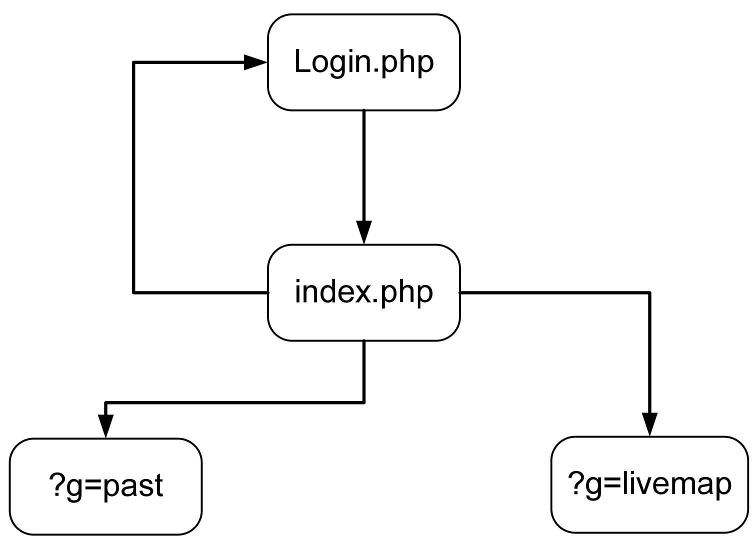


6 User Interface Design

This section provides user interface design descriptions that directly support construction of user interface screens

6.1 Application Control

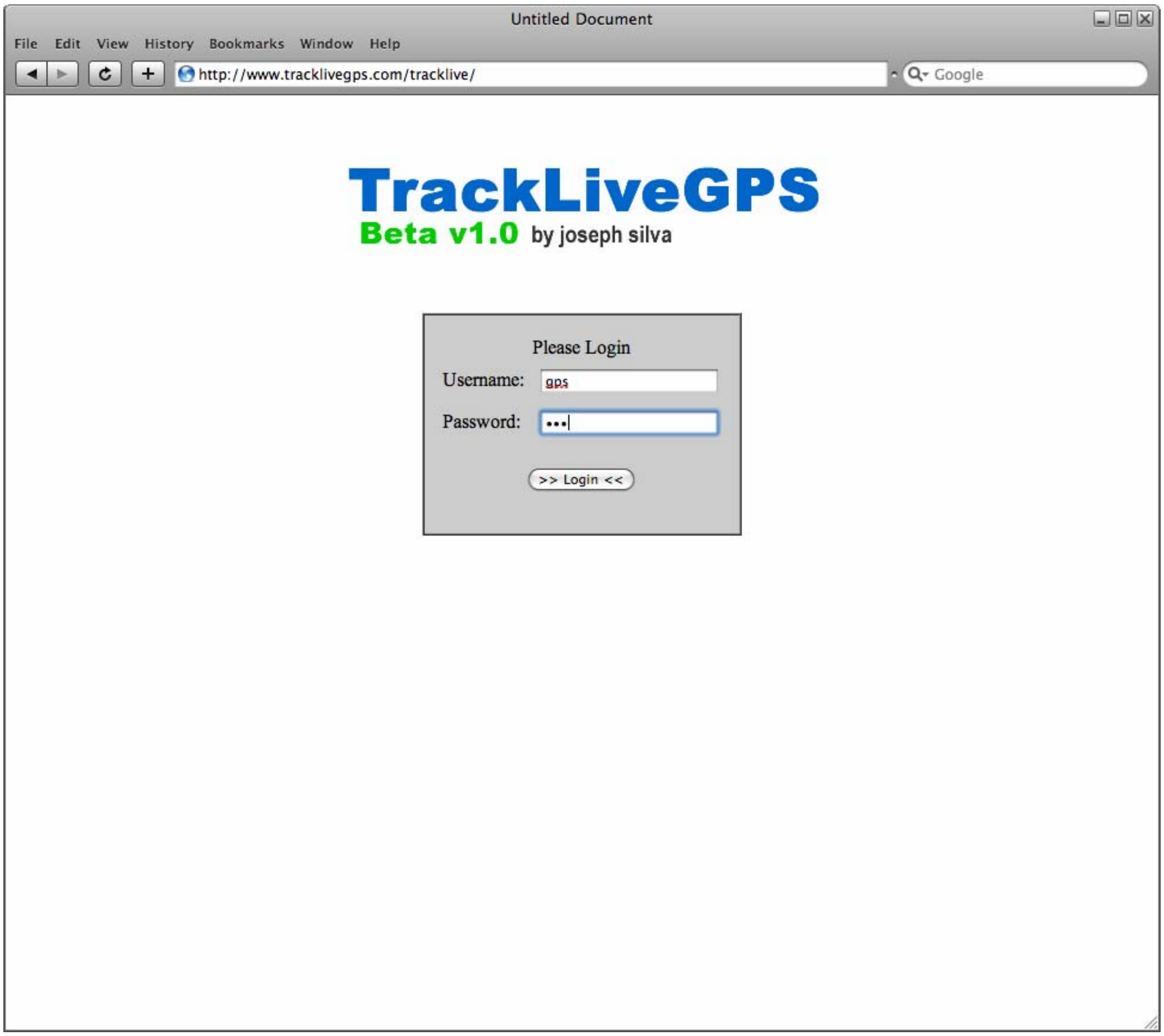
The flow chart below models the web site UI flow. When the users first visit the site they will be directed to the login page (login.php). At this point they will have to login. Upon a successful login they will be able to access the main page (index.php). From this page they can choose to view "live" tracking or "past" tracking.



6.2 Login.php

login.php:

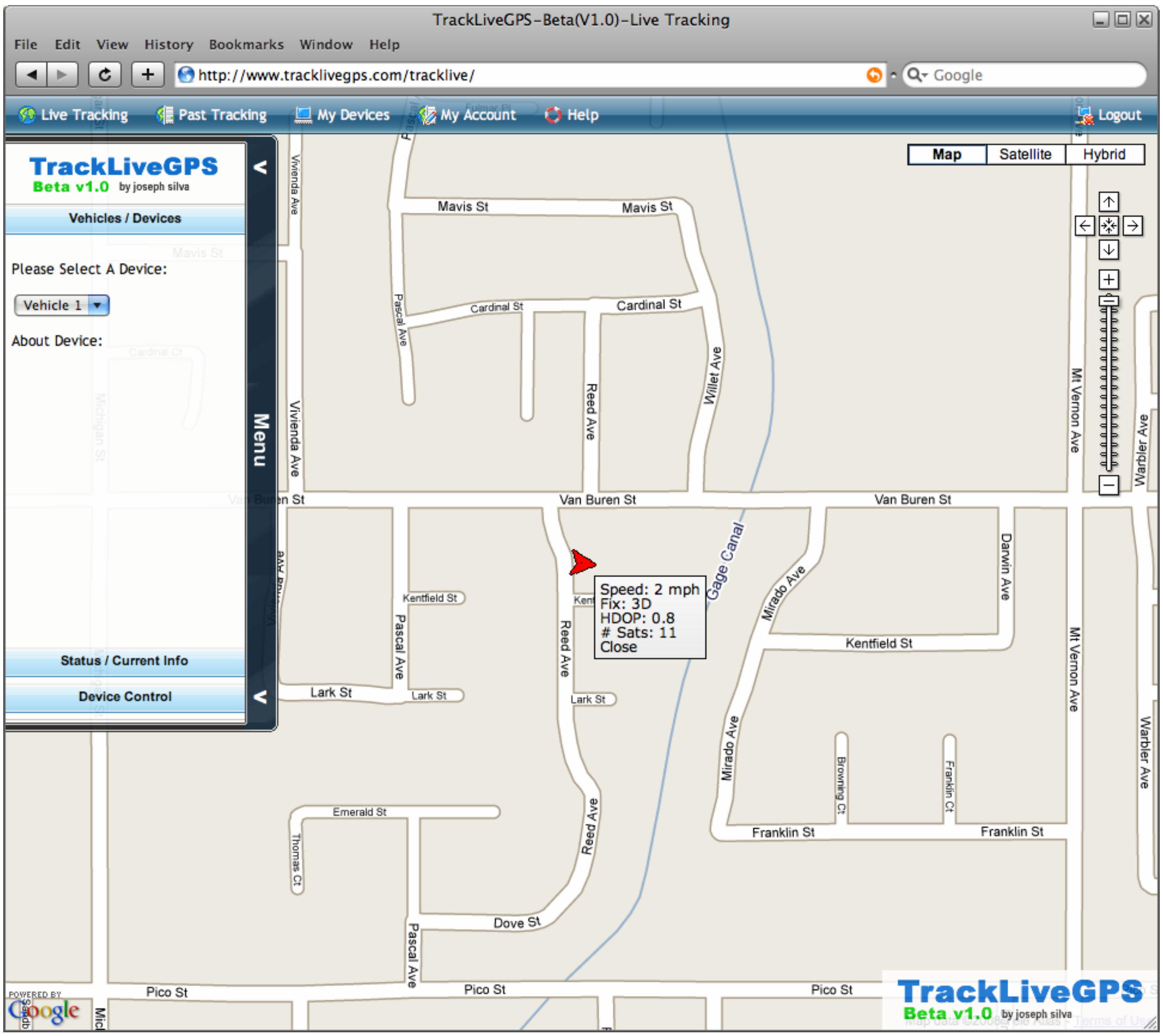
This page will handle all session creation and removal for the user interface. The users will be directed to this page if no valid session is available. The user must use this form to gain access to the TrackLiveGPS.com website.



6.3 Index.php?g=livemap

index.php:

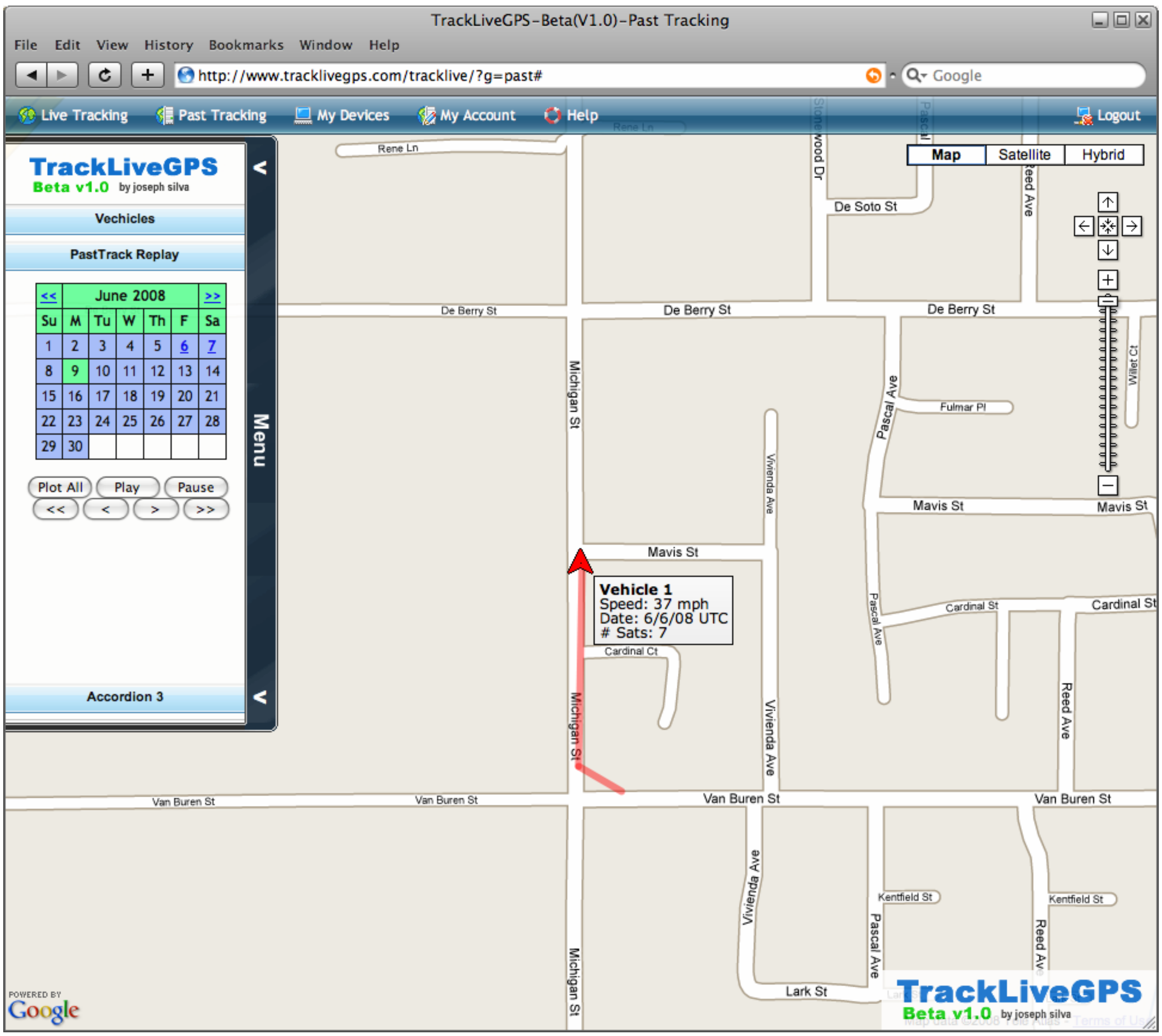
This page will by default display the live tracking data for the default vehicle in the database. The users will be able to choose which vehicles they want to track.



6.4 Index.php?g=past

index.php:

This page will by default display the past tracking data for the default vehicle in the database. It will give the users the option to select the day they want to replay from a calendar. Once the user selects a day, the system will start replaying all points logged on that day. The user will have the option to pause, rewind, fast forward, and stop the replay, as well as plot all points at one time.



7 Administrative and Other Design Issues

7.1 * Project Management

Since I was the only person working on the design project, scheduling with team members was obviously not an issue. In order to develop the project in a timely manner I set goals at various intervals throughout the design process. Each goal would build on the last accomplishment. By taking a step-by-step approach I ensured that I could complete the basics of the project first, and not get ahead of myself working on complicated features. The process that I followed was an foundation-up approach. This project gave me the experience needed to work on complicated projects in the future. By taking a very complicated system and working on one section at a time, building upon each success, the final product was possible.

8 * Experiment Design and Test Plan

8.1 * Design of Experiments

8.1.1 Test Experiment Case 1

1. Objective: Verify reliability of TrackLiveGPS AVL and VMC
2. Setup: embed three AVL devices into three different vehicles driving in southern California. Have the positional information sent every ten seconds for 72 hours (3days). Also note that for this case, even if there is no available GPS information and empty string is still transmitted to the VMC.
3. For three days all AVL devices will transmit location information. At the end of the 72 hours the database will be filled with all successful transmissions. At the end of the three days there should be 8640 transmissions. By looking at the number of transmissions on each device we can get a expectation for network reliability.
4. In order for the system to function correctly it is important that the reliability be as high as possible. For this experiment the expected reliability will be above 90% and in the area of 94-97%.

8.1.2 Test Experiment Case 2

5. Objective: Verify reliability of GPS fixes
6. Setup: embed three AVL devices into three different vehicles driving in southern California. Have the positional information sent every ten seconds for 72 hours (3days). Also note that for this case, even if there is no available GPS information and empty string is still transmitted to the VMC.
7. For three days all AVL devices will transmit location information. At the end of the 72 hours the database will be filled with all successful transmissions. At the end of the three days there should be 8640 transmissions. By looking at the number of successful GPS transmissions and thresholding the precision, we can get an estimation of GPS reliability.
8. In order for the system to function correctly it is important that the reliability be as high as possible. For this experiment the expected reliability will be above 90% and in the area of 94-97%.

8.2 * Bug Tracking

During the testing routine there were no bugs found. The system operated to spec during all test phases. Bug's only happen if you put them there!

8.3 * Quality Control

Case 1: This experiment was completed with success. We were able to obtain a steady stream of samples from the AVL devices and produce an empirical result that will be explained in section 9.

Case 2: This experiment was completed with success. We were able to obtain a steady stream of samples from the AVL devices and produce an empirical result that will be explained in section 9.

8.4 * Identification of critical components

During the testing phase, ensuring that the VMC has constant and stable internet connection is a must, or it will skew the results. Also ensuring that the devices have been installed in the vehicles in an area where they would typically be installed during normal operation.

8.5 * Items Not Tested by the Experiments

Since the AVL functions to specification, and the reliability was proven to meet the original specification, there was no need to test any other components of the system. The web interface was not tested as no real empirical results can be derived, either the system works or it doesn't. No bugs were found in the GUI.

9 * Experimental Results and Test Report

This section will provide the results for the experiments conducted in section 8.

9.1 Test Experiment Case 1

1. After conducting the Case 1 experiments it was found that transmissions successfully occurred 96% of the time.
2. This result proved to be better than the predicted values stated in section 8
3. Based on the results, %96 of the time a transmission was successfully sent every ten seconds. The other 4% account for the time when a reliable connection was not possible.
4. Since results were better than expected, no corrective measures were taken.

9.2 Test Experiment Case 2

1. After conducting the Case 1 experiments it was found that successful GPS fixes were obtained 95% of the time.
2. This result proved to be better than the predicted values stated in section 8
3. Based on the results, %95 of the time a was a valid GPS fix. The other 5% account for the time when a reliable GPS fix could not be obtained. The error could be attributed to any interference with the GPS signals.
4. Since results were better than expected, no corrective measures were taken.

10 * Conclusion and Future Work

10.1 * Conclusion

The TrackLiveGPS project completed as a success. By the time that the final project was presented, the entire system worked to the specification set forth by this document. All technical specifications were met: size, weight, cost, and performance.

The TrackLiveGPS device was proven as an effective and reliable AVL device. The device can be embedded into a vehicle and tracked live time via a web interface (up to a ten second delay). The users have the option to view all past tracking information, and replay that data.

10.2 Acknowledgements

Dr. Yingbo Hua (Section Professor)
Dan Giles
Giovanni Denina
SparkFun.com
RoundSolutions.com

11 * References

1. <http://www.gsmworld.com/news/statistics/index.shtml>. GSM Association (2007). Retrieved on 2008-05-07.

2. <http://www.gsmworld.com/about/index.shtml>. GSM Association. Retrieved on 2008-04-08.

3. Parkinson, B.W. (1996), *Global Positioning System: Theory and Applications*, chap. 1: Introduction and Heritage of NAVSTAR, the Global Positioning System. pp. 3-28, American Institute of Aeronautics and Astronautics, Washington, D.C.

12 Appendices

Presents information that supplements the design specification, including:

Appendix A: Parts List

Item	QTY	Total Cost
Telit GM862 Module	1	$140.00
Plastic Enclosure	1	$2.50
PCB (Silkscreen & Soldermask)	1	$12.00
GPS Antenna	1	$12.00
GSM Antenna	1	$9.00
Analog Components(R,C,IC)	1	$8.00
Power Cables, Connectors	1	$4.00
SIM Card	1	$6.00
TOTAL COST		$193.50

VMC (Vehicle Management Center)

Item	QTY	Total Cost
AMD Quad Core, 8GB,Raid 1 300GB, Rackmount Servers w/ Rack, Monitor, Switch, KVM, UPS, RM Keyboard. All hardware included	3	$2800.00
Windows Server 2003 Trial	1	$0.00
Fedora Core 8	1	$0.00
Visual Studio 2005, Student Ed	1	$130.00
TOTAL COST		$2930.00

GSM/GPRS Service (Per Module)

Item	QTY	Total Cost
Sim Card	3	$6.00
5MB Data Plan (9 months)	3	$8.00/Month
TOTAL COST		$18.00 + $24/month

Appendix B: Equipment List

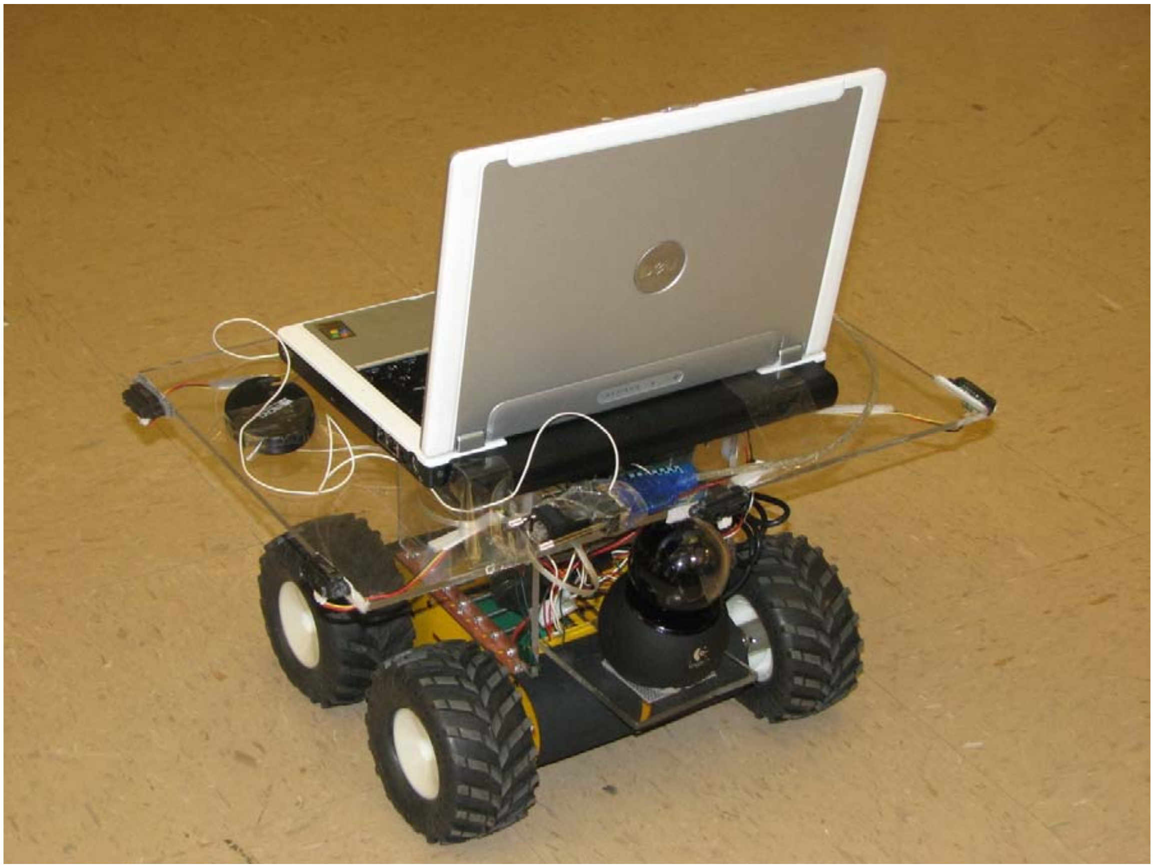
Hardware Tools	Software Tools

Telit EVK2 Development Board	Visual Studio 2005
DMM	Telit Python 1.5
DC Power Supply	Macromedia Dreamweaver
Reflow Skillet	Macromedia Fireworks
Hot Air Rework Station	Matlab
Soldering Iron	Eagle Cad
Solder Paste	Fedora Core 8
Digital Calipers	Windows Server 2003
	Windows XP Pro
	MySQL
	Apache
	PHP
	Javascript
	Google Maps API

Appendix C: Software List

Please see attached disk for all source files used in this project.

Jeeves the Robot



[redacted]

June 9, 2008

Version 1.0

University Of California Riverside
Department of Electrical Engineering

EE 175AB

Department of Electrical Engineering, UC Riverside

Advisor: Advisor: Dr. Ping Liang

Team Information:

[redacted]

Computer Engineer

[redacted]

[redacted]

Computer Engineer

[redacted]

Table of Contents

1 * Executive Summary

The overall goal of this project is to create a an entity that uses a microcontroller and PC(Personal Computer) to create a map of the of the first floor of the engineering building. Once the map is created, any user can input the location they would like to go to, the entity will calculate the shortest path to the location and then guide the user too that location taking the path it calculated. This is where Jeeves comes in; Jeeves is a robot who sole purpose is to fulfill the goals that were previously stated. In order to accomplish its goal Jeeves was divided into seven parts: 1.Mapping, 2.Motors, 3.Sensors, 4.Body, 5.Vision, 6.Interfacing, and 7.Testing. Jeeves is designed to successfully accomplish its task with the following parameters: travel at approximate speed of an average human which is 0.98 meters/second, stop within 4 four feet of the door, travel the minimal distance to its destination, transmit directions between microcontroller and PC at 1.05 seconds per direction, and should take 7 second to setup the camera after user input.

After user input Jeeves maps the path it will take to arrive at the destination that the user chose. All the destinations and intersections are represented as nodes that are connected using pointers. Each individual pointer has a specific value that represents the distance between current node and the node the pointer points to. In order to calculate the shortest path to the destination Jeeves uses Dijkstra's algorithm on the nodes. After calculating the path, the path is transferred to the microcontroller which then waits for the camera to setup. When the camera finishes its setup Jeeves begins to move, and while in transit if the camera is recognizes a door the computer will send a signal to the microcontroller. The microcontroller recognizes the signal as an interrupt and adjusts appropriately the motors and the sensors. The motors accept a frequency range to determine speed and direction, and the sensor output a voltage which the microcontroller then converts into units of centimeter. The value that is returned from the sensors is then used to keep Jeeves off of the walls and to determine if Jeeves has arrived at an intersection. Jeeves will finally stop when the camera recognizes the destination door.

The entire system (Jeeves) was tested by inputting a room number and expecting the following as output: autonomous navigation (travel in a straight line as possible), microcontroller echoing the numbers from the PC (proof that the microcontroller is receiving proper values), microcontroller signaling that Jeeves has past a door (proof that microcontroller recognizes that it has past a door), stopping within four feet of door, taking no longer than ten seconds to start. Out of all the things that were previously mentioned the autonomous movement was the only thing that did not quite meet the standards of Genaro and Ramiro. Jeeves for the most part traveled in semi zigzag pattern, and occasionally would crash into walls because the motors did adjust appropriately. On the other hand something that exceeded Genaro and Ramiro expectation was the distance that Jeeves would stop at its destination door, on average it would stop within three feet of the door.

2 * Introduction

2.1 * Design Objectives and System Overview

Our project is to design a robot that will help a person find a room at a given building. In our case the building is Engineering Building Unit 2 and we only use the first floor. The robot will use sensors to avoid hitting the walls and people while going to its destination. The user will input a room number they wish to find and the robot will show them a map and offer them to take them to the room. If the user wants to be taken to the room then the robot will then use a camera and dijktras algorithm to help navigate the robot to its destination. We are using a laptop to do all the image processing and upper level software decisions while a micro-controller is doing the lower level decision to make the robot move and avoid objects.

This project fits well in the field of robotics and autonomous systems in the Electrical Engineering field. It's a major research area and new technology has come out from it. We have smart vacuum cleaners, smart vending machines and other products that user the subject of robotics and the subject of automatic. Our project could be a very helpful object in buildings that are hard to navigate. The robot could help in hospitals, schools, government buildings, or other large buildings that are sometimes hard to find places in.

The robot had two main parts that do all the work. One part is the laptop that will be in charge of doing all the image processing and the mapping for the first floor. It will provide a path to the microcontroller and to the camera in order to detect the doors. The second main part as mentions is the microcontroller. This will be in charge of using the sensor to avoid obstacles and to make the robot move. The laptop will communicate via serial with the microcontroller providing it the path that it needs to follow and by giving it signals to signify a door was detected and a stop signal.

Some of the design objectives for our project are as follow:

- speed of robot should be about 0.98 meters/seconds
- Communication computer and camera to micro-controller should be real time. Less than a second
- We should be using at least 3 sensors but maybe more.
- Major objective is to have the minimal distance to travel.
- Total price of material should not exceed 500 dollars (actual expenses 700)

Responsibilities:

- Ramiro Diaz (Project Manager)
  - Make sure objectives are being completed in time
  - Manage the project
  - Computer Vision
  - Mapping
  - Combining all code together
  - Debugging
- Genaro Velasquez
  - Sensors
  - Motors
  - Mapping
  - Autonomous code
  - Debugging

2.2 * Development Environment and Tools

We used Freescale CodeWarrior for HC12 version 5.7.0. We used Microsoft Visual Studios 2008 for writing most of the camera code and combining all the other code together. We also used cygwin to do some of the coding for graphical purposes. Some of the tools used were a hand drill, multimeter, power supply, and oscilloscope.

- Freescale CodeWarrior for HC12 version 5.7.0
  - http://www.freescale.com/webapp/sps/site/homepage.jsp?nodeId=012726
- Tera Term Pro Version 4.55
  - http://ttssh2.sourceforge.jp/
- Microsoft Visual Studios 2008
  - DirectShow Pan/Tilt/Zoom sample for Logitech QuickCam devices
    - http://www.quickcamteam.net/documentation/how-to/how-to-control-pan-tilt-zoom-on-logitech-cameras
  - Opencv
    - http://sourceforge.net/projects/opencvlibrary/
  - C++ coding
- Cygwin
  - ccc_win libraries to use graphics
  - C++ and system calls.

2.3 Related Documents and Supporting Materials

Industry Standards

- USB
- RS232
- ATD
- C & C++
- PWM
- SCI

2.4 * Definitions and Acronyms

- GP2D12: Sharp IR sensor used for keeping robot of the walls and to make it autonomous.
- PCB: Printed Circuit board designed by the team and printed on a Mill.
- MC9S12DP256: Motorola microcontroller used for making the robot move.
- USB: Universal Serial Bus is used for the camera and for the serial to USB connections to micro-controller
- RS232: This is the connection that connects the microcontroller to the computer. It also uses the USB to accomplish this since our laptop didn't have an RS232 connection.

- .s19: This is the file that has the information for the microcontroller. It tells the microcontroller what to do and how to do it.

3 * Design Considerations

3.1 * Assumptions

1. Jeeves starts at a fix location that Ramiro and Genaro have titled HOME in the code. Jeeves needs to begin at this location because it has no other means of telling where currently is at. If Jeeves does not start at that fixed location it will make the correct turns and stop, but the location that it stops will not be the desired destination.
2. All the destination doors have a neon green color that helps Jeeves recognize that it is a door.
3. The environment in which Jeeves is traversing is properly lighted especially were the doors are at.
4. The computer code is runs under Windows XP, and the microcontroller code runs using the MC9S12DP256.
5. The laptop used weighs no less than 4.5 pounds; otherwise Jeeves will not be able to turn.

3.2 * Realistic Constraints

Money was one of the main constraints that we faced. It held us back from buying better sensor that would have helped with the intersections, better motors that will carry more weight, and from buying a lighter laptop. The next constraint that we encountered was the ideal environment. The ideal environment is one in which the halls are well illuminated, there are no obstacles, and the ground is not at a slope. The next constraint was power, since Jeeves needed to travel long distances the power needed to be portable. The power for the motor needed to be at least 7.2 V at 3300mAh, the microcontroller needs a 9V battery, and the sensors need at least 5V. The processor needed to be at least 20 MHz to handle motors, sensors, and SCI.

3.3 * System Environment and External Interfaces

The main software for the computer is runs in Windows in XP--> Visual Studios, on a Dell Inspiron m133 with an Intel Centrino M. The main software took care of five main things: map, sending path, sending interrupt, camera vision, and camera pan. The mapping software displayed a GUI for the user and accepted and remained idle into the user inputted a correct room number. After the user has inputted a correct room number the mapping software displayed a path and then outputted it to the terminal. The main software would then redirect the output to a file. The main software would then parse the file into single values and send them to the microcontroller. In order to send the values the software uses SCI and the local COM ports. Once the entire path has been sent the main software would setup the camera. If the camera recognizes a door the main software sends an interrupt to microcontroller. Depending on current location, the camera pans so that it can recognize the next door. For example if the door is located on the left the camera pans to the left before hand so that it can be ready to recognize the door.

The MS9S12DP256 microcontroller code is part of the HCS12 family. The microcontroller took care of the following: motor, sensors, and SCI. The motors in combination with the sensors provided Jeeves with autonomous movement. Every 4.096 ms the microcontroller would read the sensors and store the values into global variables after having passed through a filter that throughout the random values. The microcontroller then used those values to output the frequency for the motors and the frequency determined the speed and the direction of the microcontroller. The microcontroller used the SCI to download the path from the PC beforehand.

3.4 * Industry Standards

- ATD – used for the sensors to convert voltage into a digital value.
- RS232 – used to communicate between microcontroller and PC, and for interrupts.
- USB – used to communicate between microcontroller and PC.
- C – used for microcontroller code.
- C++ – used for PC code.
- PWM – used to control motors.
- SCI – use to send path.

3.5 * Budget and Cost Analysis

The budget that Genaro and Ramiro set is $500 dollars. The actual cost was

$25.95	-7.2Battery charger
$14.95	-7.2 Ni-MH Battery
$64.99	-Sabertooth Dual Channel Motor Controller
$ 4.95	-Batter Connector
$27.26	-Delivery Charger
$ 3.99	-Plastic cutter
$14.47	-Plexi Glass
$ 4.23	-Tax
$23.68	-9V Battery
$ 4.83	-Paper
$29.92	-Velcro, Screws, and Tape
$113.11	-Logitech Orbit Cam
$246.76	-Lynxmotion 4WD2 Chassis
$579.24	-TOTAL

3.6 * Safety

In terms of safety there is nothing on the robot itself that can harm humans or other things. The only thing that might harm somebody is when the robot bumps into someone or something. Genaro designed an autonomous code that will help prevent this situation.

3.7 * Documentation

Each team member held their own notes and maintained their own data for each experiment ran. When it came to version control each team member updated the part they were working on as they saw fit.

3.8 * Global, Economic, Environmental and Societal Impact

Jeeves the butler type robot will take in input from a person, either typed or (hopefully) spoken, and use this input to give directions to the user to the location that they chose. After which if the user desires Jeeves to take him or her to that location then Jeeves will do so. Once Jeeves has completed it task it will go to the closest of one of three predefined stations and wait their idle until it is given new input from a user. It is this simplicity in use that will make Jeeves one of the greatest localized navigation tools available on the market (assuming that it is mass produced), not only that Jeeves combines an information kiosk and a butler into one simple package. Having what can easily be called information on the go will be huge time saver for the user, especially people like freshmen who most likely will enter a building and it will be their first time in it, they will need something like Jeeves to help them get around. This brings us to the type of people that will use Jeeves, the type of people range from a businessman in a corporate office to a visitor in a hospital. Anybody wanting directions to a room within a certain building can be helped by Jeeves.

Considering that Jeeves can help anybody the quality of life of any person in an office, building ...etc can be improved by making some simple modifications to the original version of Jeeves. All that would simply need to be done is to make Jeeves carry more weight by using motors that can output more torque and replacing the battery with a bigger one. If Jeeves can carry more weight the user will be able to have Jeeves carry any form of luggage for the user, making the user less tense and less tired by the time they arrive at their destination. Although Jeeves can provide the user with a couple minutes of relief on the way to the destination the user wanted, it does not provide much in the entertainment department aside from playing some music on the way to the destination. Even though Jeeves does not have many uses for entertainment, it can used to educate the user about the building. Jeeves will then provide facts about the building particularly about certain markers as they pass by them.

Jeeves has the potential of doing a lot more than just giving directions and having the user follow Jeeves to the destination they chose. For example the last sentence of the previous paragraph hinted at having Jeeves run on a routine from one room to another meanwhile giving out facts about the building as they roll along the entire building. Assuming that the code update is correct then we will have a Jeeves version that can serve as tour guide. The main problem with having Jeeves serve as a tour guide and with Jeeves the original is that many jobs can be easily replaced by Jeeves, effectively lowering the number of jobs that will be available in these fields. On the other hand jobs in the engineering, manufacturing, and maintenance of Jeeves will rise. Which means that the jobs that can be classified as blue collar will diminish and make white collar jobs rise, this will make it harder for people who are not trained or properly educated to get jobs. Considering that Jeeves can be easily adapted to be used globally in any place that can be considered an office environment, the economy, well at least the job aspect, can affect the entire world as more places adapt Jeeves into their environment.

Regarding the environment, Jeeves will be composed of materials that are readily available in the market meaning that pollution will be the same as that of the materials already being created. Aside from having the production of Jeeves create more of a demand for these materials Jeeves will not add any extra pollution to our environment. Considering that Jeeves will not add any more pollution to our environment there should not be any political debates regarding the production or use of Jeeves in everyday life, and aside from some minor patent and more business related issues Jeeves should not have any problem with adhering to the laws of any country. On the other hand their might some social issues that might need to be dealt with as they arise. Jeeves like many other new products will eventually make people socially retarded. These people get so accustomed to talking to a machine that when they are put in a social situation they have no idea what to do, and unfortunately Jeeves will add to this problem.

3.9 * Contemporary Engineering Issues

Issues that revolve around Jeeves can be classified into following: potential uses for Jeeves, industry standards, power, and materials. Before going into the issues, it is necessary to understand the essence of what Jeeves does. Jeeves main purpose is to receive a destination and then go to that destination using onboard tools to help it navigate the terrain. Among the tools that Jeeves will use are sonar sensors, infrared sensors, treaded wheels, DC motors, odometer, a topological map, and a camera. The purpose of these tools is so that Jeeves will not get stuck on the way to its destination. The path that Jeeves chooses must be shortest path, especially since Jeeves will most likely be guiding an actual person to the destination that was chosen.

Here is where one of the issues regarding Jeeves arise, the one about the potential uses of Jeeves. Considering that Jeeves moves from one destination to another, the uses for Jeeves can vary and not all these uses will always follow Jeeves original goal of taking the shortest path possible to arrive to at that destination. For example the current model that Ramiro and Genaro are building is made so that Jeeves will receive an input from the user, and then use this information to calculate the shortest path to the destination that the user chose. This version of Jeeves can be used at a school or a building or any place for that matter that constantly has people who are lost and need assistant finding the room they need to go to. Another version of Jeeves can be designed to conduct tours of a building or a campus, which essentially is having Jeeves run on a subroutine that makes it move from one destination to another while giving some facts about the place that it currently is at. The location of the beginning of the tour and the path that it takes to arrive to its ending location will not always be the shortest. This can become an issue in terms of programming and manufacturing because then major changes have to be made, because the path that it will take is in direct violation of Jeeves original goal.

In order to change Jeeves goals, there must be some standard that Jeeves needs to use, and there is no real standard that the industry can use, because every situation requires a different approach. For example if a version of Jeeves was created to carry items from one location to another, obviously a different set of priorities must be used because the extra weight will make Jeeves consume more power. Considering that Jeeves will use a different set of priorities there is no real standard that can be defined that will fit every situation. Although this situation does not have a standard for it, there are other standards that Jeeves will use. For example the sonar sensor will use the Inter Intergraded Circuit (IIC) standard, the infrared sensor will use the analog to digital conversion, and the microcontroller will use the HCS12 to program it. The problem comes when we have to change a part for a different one. For instance the sonar sensor, this sensor sends out a series of supersonic pulses and then records the time that it took to echo. The problem with this is that not all buildings are the same, so different materials used on the walls may cause the sensor to never pick up the echo, which means that Jeeves needs a different way of telling the distance from the wall, possibly meaning that IIC will no longer be used, and may even require a completely different microcontroller. Now if the microcontroller is changed then we need a different way of controlling the other sensors and motors, which essentially is like redesigning Jeeves, which can be costly.

Taking into account that redesigning Jeeves can be costly, the materials that Jeeves will be composed of are already, a bit on the expensive side. For instance the platform that Jeeves will use is sixty dollars plus another twenty for the driver for the DC motors; so far it comes out to eighty dollars. Plus the more expensive materials like the microcontroller ($120), the sensors ($70 each), the camera ($50), plus display and computer ($500), and other production cost. Easily Jeeves can end up costing around $1000 dollars plus the cost of possibly having to redesign it. Of course certain things like the microcontroller can be changed for another less costly microcontroller that is more specific for the applications that Jeeves will use it for. Eventually interchanging some of the materials can make Jeeves less costly but not enough that “everybody” would like to buy one. This is where the concern of materials comes in, because just to

4 Experiment Design and Feasibility Study

4.1 Experiment Design

Describe the objective, setup, procedure and expected results of each experiment.

- Camera Testing

  For this test we are going to be making sure that the camera pans to the right directions that the door is located at. Then the camera will detect pixels within a certain range of RGB colors.

  1. To test first we need to have some sort of marker on the doors and then have a laptop with the camera so we can run a test on the pick up of the color on the markers.
  2. Next is to run the program we created and make sure that the output is as expected.

  The expected results are to receive a 1 if the door was recognized and 0 if the door is not recognize. The program we created is also printing to terminal whether it's a door or not.

- Sensors

  This involves testing each sensor and multiple sensors at the same time. We created code for multiple sensors to work at the same time since that's the main purpose is to make the robot autonomous.

  1. The first thing to do is to make sure that all the sensors are correctly connected to the bread board. The power that each sensor uses is 5 volts.
  2. Next we ran the code we made for this test and made sure that the program is really reading each sensor.

  The output of the test is to the terminal of tera term. It will display each sensors reading and output it in millimeters to the terminal.

- Serial Communication

  The way we test the code we created for serial communication is by also making code for the microcontroller that expects the value or data sent by the serial code and outputting it to the 7 segment LED.

  1. First thing to do is create a testing program for the microcontroller and make sure it works.
  2. Next is to run the executable serial with an argument which is a digit in the range of 0 to 9.
  3. Next read the output of the 7 segment display.

  The output of the this test should be the same number you sent with the program serial.

  Example: $ serial.exe 1   expect 1 on the 7 segment display.

- Autonomous Navigation

  This is to test whether or not the robot will stay away from objects and not crash.

  1. First of all is to have the code for the microcontroller loaded onto it.
  2. Then make sure the battery is completely charged because the results of the test could be different depending on the amount of power the battery has.

The outcome of this test should be that the robot does not hit any wall and it does something if the robot does sense a wall or object in any view of the sensors.

- Software testing

The software testing involves the testing of each program individually. Next would be to start combining code and testing it as we keep adding it.

1. First we tested everything individually.
2. Next we made sure we had an idea on how everything was going to fit in the big software diagram.
3. Next we added code and tested and fixed any bugs that came up.
4. finally put it all together and come up with solutions to the problem that come up. We do this until we get the entire system software to work.

The system software can be expected to only have one output. That is to have a working product at the end.

- Testing of entire system

This test involves putting all the individual modules together and testing them as a whole. The system when combined can act very differently and may not be to our satisfactions. Therefore the test needs to be taken very serious even thought we have already tested the modules individually.

1. First of all is to make sure that batteries are fully charged and that the hardware is working properly.
2. Next test the individual modules one last time and make sure they are still working.
3. Next run the complete system and see if everything funs smoothly.

The system is expected to run autonomous and the camera is expected detect the door where there door is seen. Then if something goes wrong just debug it and run the experiment again. At this point its all about trial and error.

Responsibilities for testing:

- Ramiro Diaz
  a. Camera testing
  b. Entire System testing
  c. Autonomous
  d. Software testing
- Genaro Velasquez
  a. Sensor testing
  b. Serial Communication testing
  c. Entire System testing
  d. Autonomous testing

4.2 Experiment Results and Feasibility

Carry out the experiments designed above and present experimental data, quantitative analysis of the data, and the conclusion to show the feasibility of your project idea, how the experiments help you decide whether your technical design objectives can be achieved, and how they help you select the best solution to be further developed in the design project.

- Camera
  The testing for this module turn out somewhat problematic so we had to change the way we were detecting the doors. Instead of using the entire image we only concentrate on some pixels and find that one that matches the range and return a 1. This change was very successful on the testing and thus we made the change in our plans. The camera responded as expected and made our project a success.

- Sensor
  The sensors were causing us some problems and we had to really spend some time debugging them. We had problems with the power and had to group some sensors together and power them in groups. This solved that problem. Then we had problems with making all the sensors work together. But finally we did get the test to give us really good results.

- Serial Communication
  The results of this were very positive except when trying to call the executable from the Map program. So we ended up having to write to a file and have the main code make the call to the serial executable. The numbers that were transferred from computer to microcontroller were showing up on the seven segments LED that is attached on the microcontroller.

- Autonomous Navigation
  For this part we had to get the sensor testing done first and made sure they worked at least 90 percent. After that we got some really good test runs and made some adjustments to the code so that it accommodates for some wrong readings.

- Software Testing
  This was of course very successful. The only way to have the system work is to have the software work. We did have problems with merging all the software and got experience with debugging it. Finally we got it all to work and had the testing of the entire system ready for testing.

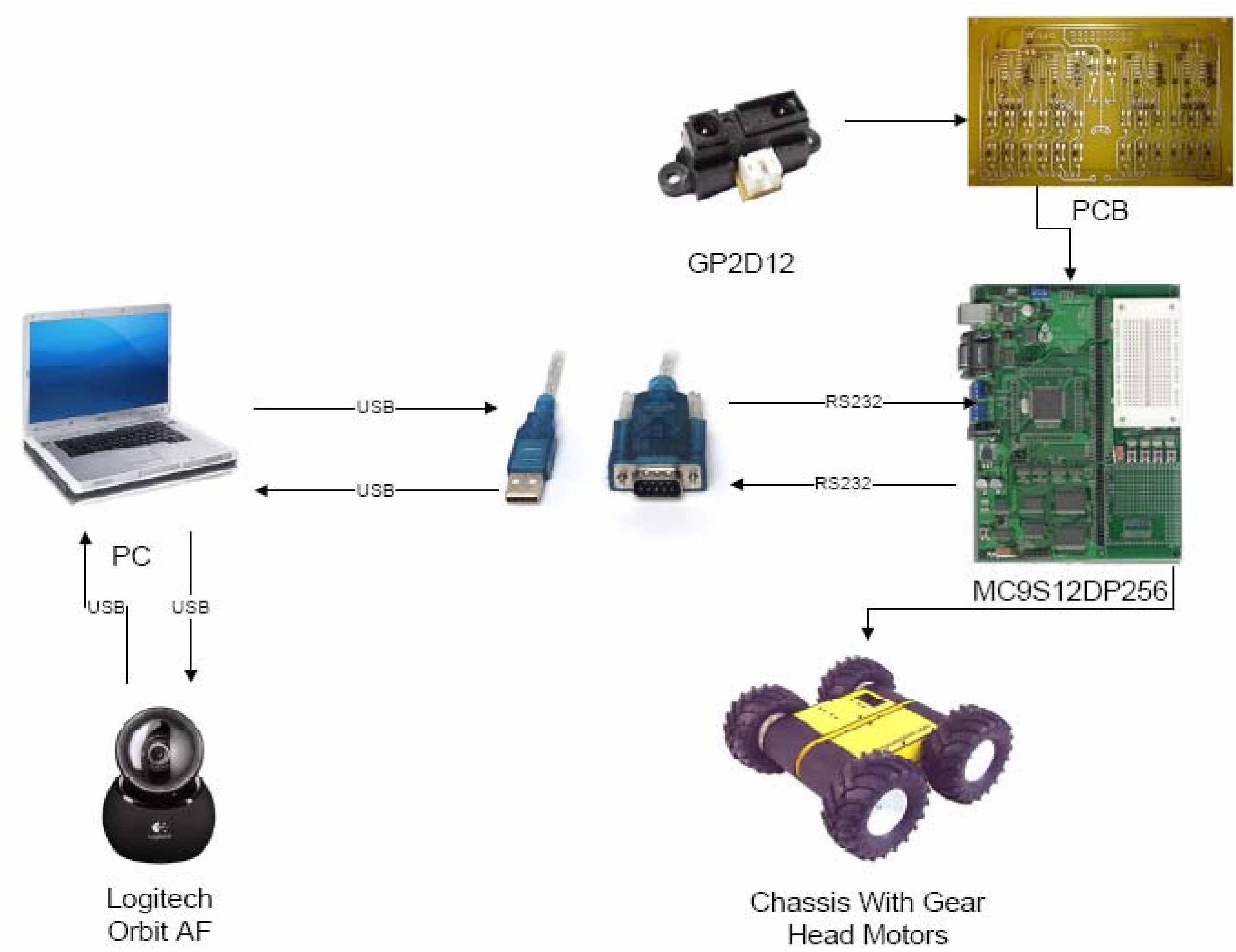
- Entire System
  The entire system gave bad test runs at the beginning but as we kept fixing bugs and changing some designs we kept improving the test runs. We finally got some really good runs and got them on video. The system did work correctly and we are happy that we did test everything individually first.

Responsibilities:

- Ramiro Diaz
  a. Camera testing
  b. Entire System testing
  c. Autonomous
  d. Software testing
- Genaro Velasquez
  a. Sensor testing
  b. Serial Communication testing
  c. Entire System testing
  d. Autonomous testing

5 Architecture

5.1 System Architecture



The hierarchy of the design is the following:

Laptop Camera

Microcontroller sensors

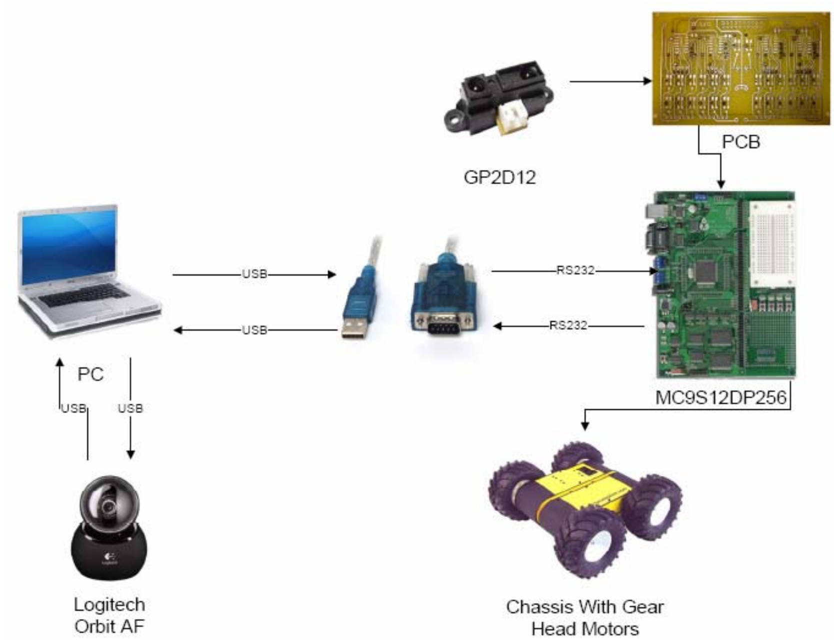
4WD2 base

5.2 Rationale and Alternatives

The architecture that we are using is what we had planned from the beginning. We have that kind of architecture because it's the best arrangement that could give us the best results. The camera needs to be able to detect a door and the only way that would be possible is if we have a computer to do the processing and to make sure its quick enough. The laptop is there also to communicate the path to the room number to the microcontroller. So the laptop is taking care of the most power and processing consumption jobs. The laptop is connected to the microcontroller via a serial to USB and RS232 connection. The microcontroller takes readings from the sensors and from the computer to tell the robot what way to more and where to go to. The microcontroller also makes the motors to work and thus the 4wd base is the bottom layer of the hierarchy.

6 * High Level Design

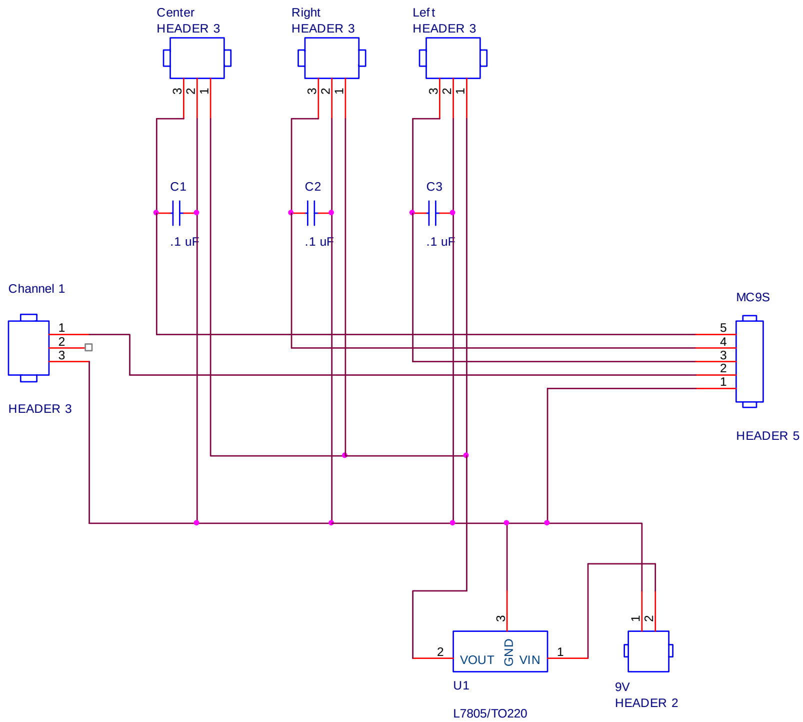
6.1 Conceptual View



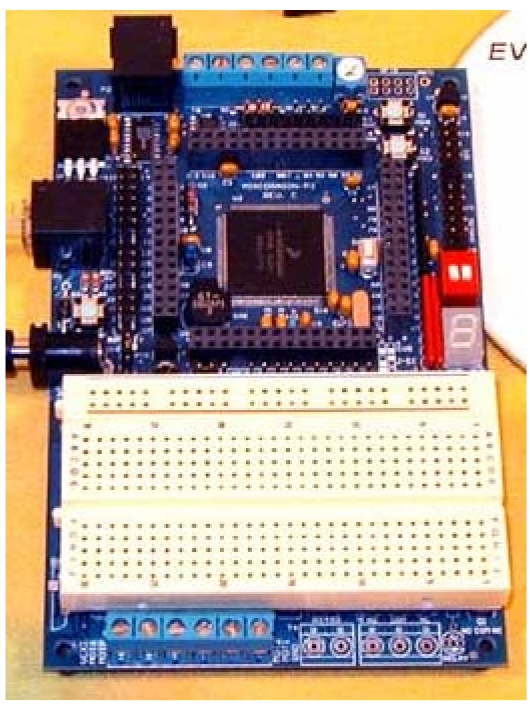
This is basically the idea of the project and how its going to be organize. The system is also concentrated a lot in software.

6.2 Hardware

The sensors we are using are the Sharp GP2D12 IR sensors.

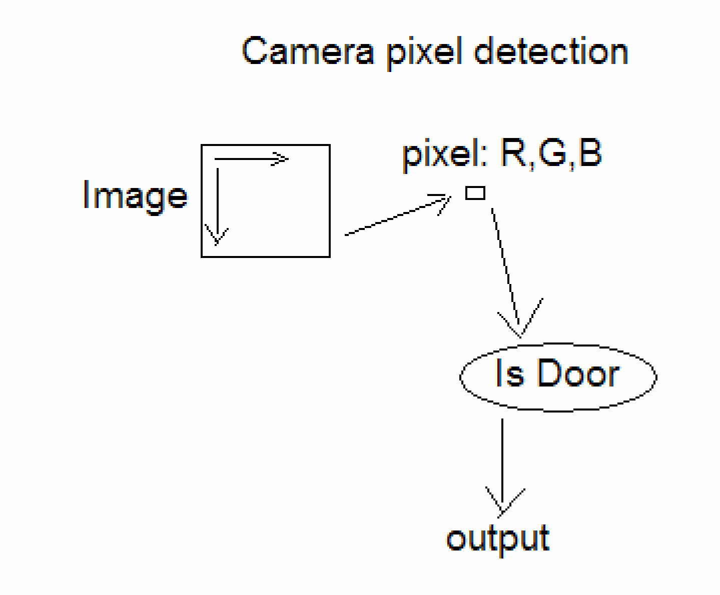


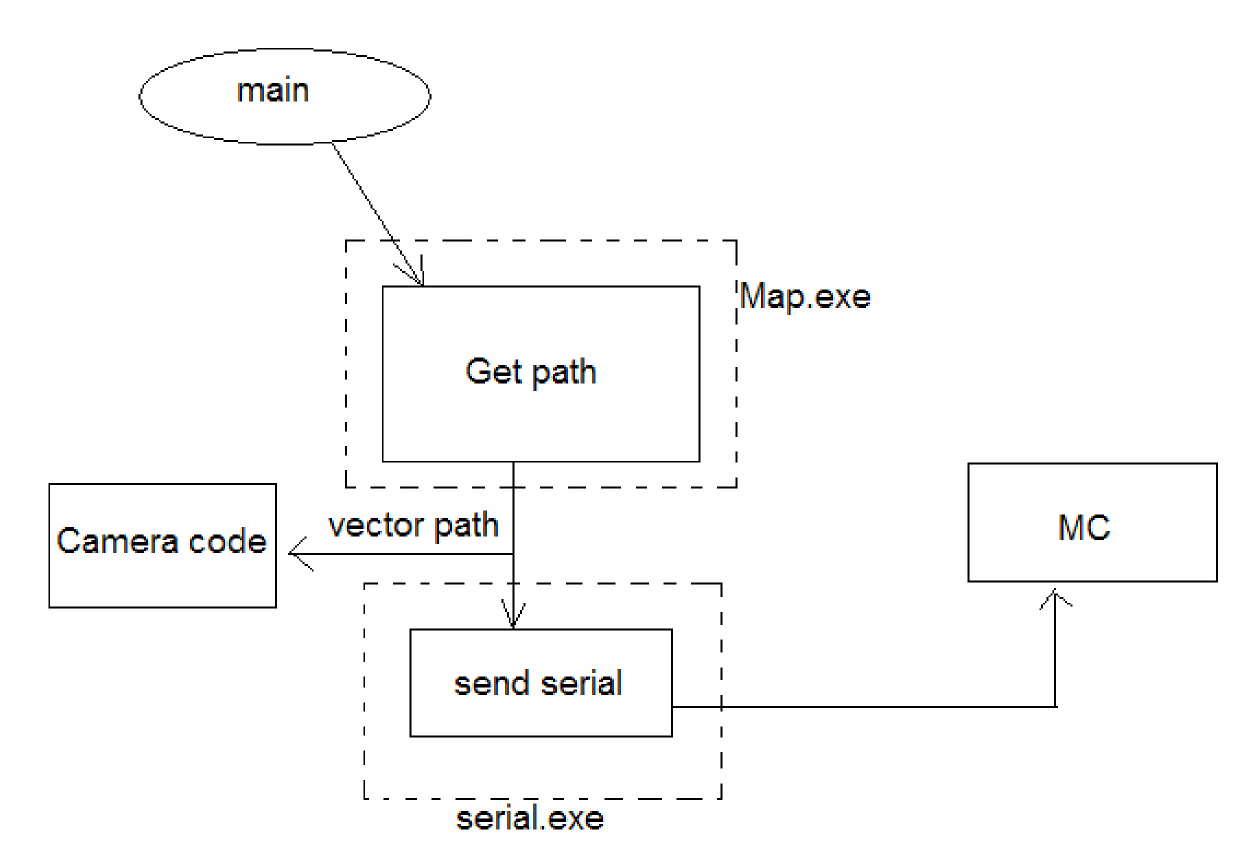
Then we have the sensors connected to the microcontroller.



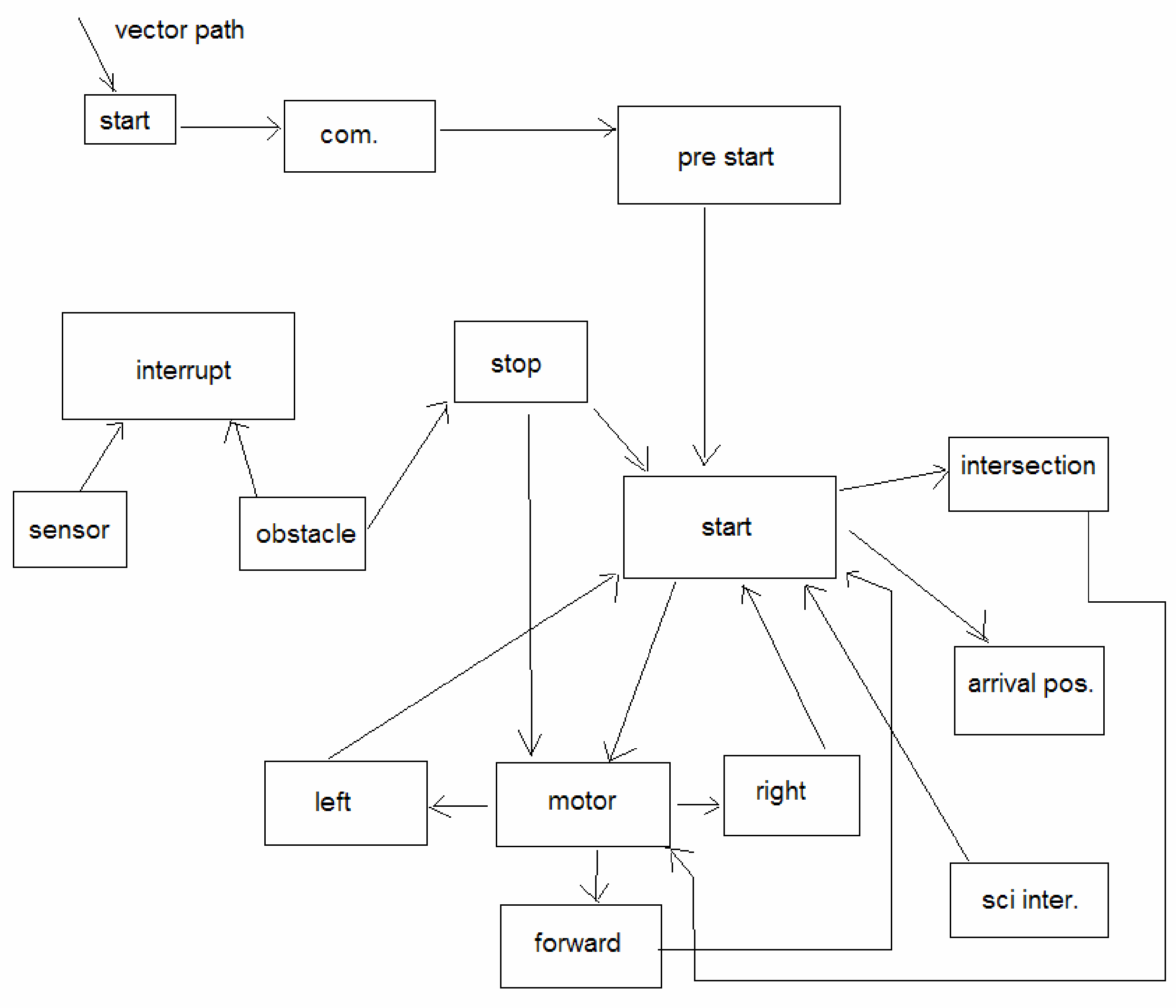
The microcontroller has the motors of the 4wd base attached to it and that would create the lower level. The next level would be the high level which includes the laptop and the camera. That in itself is doing basic stuff that doesn't have much hardware details except for the usual.

6.3 Software


Camera pixel detection
Image
pixel: R,G,B
Is Door
output


main
Map.exe
Get path
Camera code
vector path
MC
send serial
serial.exe

This is the main microcontroller path of the program. This is mainly what happens every time the micro-controller gets a path to execute.



7 Data Structures

7.1 Internal software data structure

Some of the internal software data structures that we used were vectors and arrays. We also used Nodes and a linked list that makes a map of the engr2 building. The vector acts as a way to store the path of nodes and the array serves as the container to store the path on the microcontroller level.

7.2 Global data structure

The major global data structure is the Node and linked list. They are available to all the other pieces of software. It's the main medium of the generating a path to the room number that the user inputted.

7.3 Temporary data structure

The file that is created for the use of the program is only one. It's the file that is used to read the information to get to a room number. It was a way to solve one of the problems that came up when joining code from two different platforms.

8 * Low Level Design

8.1 Module 1: Camera Code

8.1.1 Processing narrative for module 1

The Camera Code checks every pixel and looks for the RGB values that are within the parameters that we set. The code then uses the pan option and uses it to turn to the upcoming door. Once the camera code is started a constant feed is read in and processed. Once the code finds a pixel the fits the criteria that Ramiro has set the Camera Code exits and outputs a one, which signifies that a door is read.

8.1.2 Module 1 interface description.

The video feed is read in from the USB port. The port is then connected to the Logitech Quick Cam Orbit AF. The video feed is read in at 30 frames per second and at a video quality of two megapixels. The Camera accepts input for zoom, pan, and tilt. The camera code uses this device to read in the information from the environment and once the proper combination of pixels is found the camera codes exits and out a one to signify that a door has been found.

8.1.3 Module 1 processing details

The Camera Code stores every pixel of every frame in a vector. Then it checks the R,G,B values of every pixel and makes sure that sum adds up to color of neon green.

8.2 Module 2: Map

8.2.1 Processing narrative for module 2

The map software uses hard coded points to draw a map of the terrain, when finished it creates all nodes that represent either an intersection or a room. Map further goes on to connect each node like with pointers. After the user the has inputted the destination room Map will use Dijktra's algorithm starting from HOME. After calculating the path the path is outputted to the terminal one by one, while outputting to the terminal it uses serial1 to transmit the value to the microcontroller. When finished it displays the path on a GUI were connected dots show the shortest path to take to arrive at the destination.

8.2.2 Module 2 interface description.

The small GUI that the code displays is created using CCC_WIN library. The GUI uses set points that Ramiro and Genaro calculated and measured by physically going around the first floor of the Engineering 2 building in UCR. The GUI will then wait until the user inputs a correct room number, if a correct number is not inputted then it just sits there and waits until one is. The shortest path is then displayed on the screen as dots and lines that connect those dots and on the side of the map numbers show up that represent either a left or right turn. As the number are the displayed on the GUI they are also outputted to the terminal.

8.2.3 Module 2 processing details

Once the proper value is inputted a digital map of the floor is created using nodes. The node are then connected with pointers which also reference a float value which represent the distance from the current node to the node that it is pointing to. Dijktra's algorithm is then called on this collection of nodes, when the algorithm is finished it returns a vector of the nodes to take in reverse order.

8.3 Module 3: Serial

8.3.1 Processing narrative for module 3

Serial will take in a value passed in from the terminal, the values is then converted into ASCII. Once the is set the value is passed into MySerial.h code. MySerial takes the value and outputs to a hard coded COM port which in Jeeves case is COM2.

8.3.2 Module 3 interface description.

The code itself is an executable that will only run under Windows and Linux. The pixel code calls serial when it wants to send a value to the microcontroller and when it wants to send an interrupt signifying that it has recognized a door. When sending a value to the microcontroller serial will transmit the value at a baud rate of 9600 with no overflow bit.

8.3.3 Module 3 processing details

Serial uses the USB COM port to send information out; the only constraint with this module is that only one character can bet sent at a time.

8.4 Module 4: GP2D12 Code

8.4.1 Processing narrative for module 4

This code uses the ATD ports of the microcontroller to convert the analog values that come in from the GP2D12 IR sensor in to digital values that are between 0-1023.

8.4.2 Module 4 interface description.

Code the reads in the values that are stored in the in the ATD registers. The value that is read in is between 0-1023, and then in it is converted into centimeter units using this equation **R = (6787 / (V - 3)) – 4 .** The output is then a centimeter value that is stored into global variables.

8.4.3 Module 4 processing details

The values that are stored into the ATD registers are stored with a ten bit resolution and unsigned. The code reads the values from six ports consecutively from ATD1.

8.5 Module 5: Motor and PWM

8.5.1 Processing narrative for module 5

The microcontroller uses PWM modulation to control the speed and direction of the motors. The PWM is given a value between 100 and 200. Then closer value gets to 100 the faster the motors move in reverse and the closer it is to 200 the faster they move forward, and when the value equals 150 the motors will not move.

8.5.2 Module 5 interface description

Values are given to the PWM based off the values of the sensors and the directions that Jeevees wants to go to. Only two wires are connected from the PWM ports to the drivers one for the left side and one for the right side. The motor drivers is then connected to left side which has two motors connected in series and to the right side which follows the same regimen as the left side.

8.5.3 Module 5 processing details

PWM outputs a frequency and then the motors move.

8.6 Module 6: Communication Protocol and Interrupts

8.6.1 Processing narrative for module 6

This protocol basically sits idle until the entire path is downloaded from the computer. The first value that is downloaded from the computer tells the microcontroller how many turns to make. Based off of the first value of the protocol, the microcontroller waits until all have been downloaded and then proceed to move.

8.6.2 Module 6 interface description.

The values that are downloaded are expected to be to come from the SCI0 port of the microcontroller. The values are then stored in the in an array and used to navigate the terrain.

8.6.3 Module 6 processing details

The values that are downloaded are expected to help Jeeves navigate. This information is very crucial for the project considering that without Jeeves is lost. When an interrupt occurs it tells Jeeves to move on to the next direction in the array.

9 User Interface Design

This section provides user interface design descriptions that directly support construction of user interface screens.

State clearly who is responsible for which module/task

9.1 Application Control

Detail the common behavior that all screens will have. Common look and feel details such as menus, popup menus, toolbars, status bar, title bars, drag and drop mouse behavior should be described here. Conventions and standards used for designing/implementing the user interface are stated.

9.2 Screen 1..m

Illustrate all major user interface screens and describe the behavior and state changes that the user will experience. All screen objects and actions are identified.

A screen transition diagram or table can optionally be created to illustrate the flow of control through the various screens.

This does not have to be actual screenshots since they have not been programmed. They can be power-point drawings or mockups created in Visual Basic or some other rapid GUI-building tool.

9.3 Development system and components available

The user interface development system and GUI components available for implementation are described.

10 Administrative and Other Design Issues

10.1 * Project Management

One paragraph from each team member
How was the project managed, how was tasks distributed, how was scheduled made, what project management software/method did you use, your experience in working together, what have you learned in terms of team work, time management, project management etc., from this project.

Ramiro Diaz:

The project was managed well. Both team members had a say on the schedule and we both helped each other and were there for each other. We were not organized in terms of having a schedule written down or using any project management software. We did try to plan for a weeks worth of tasks. But we mainly concentrated on and came up with objectives as we saw the need to. Working in a team has taught me a lot. First of all I learned that we should all be able to understand what the other team member is doing and be able to ask for code or schematics of design and be able to read them and know what they are doing even though we did not write the code or design the circuit. I would say that was our main weakness as a team. We did not work on code together unless it was debugging or helping with certain problems. In a way it’s good that we don’t interfere with each other but we should keep in mind that the other person should be able to read any comments we have in code or notes on whatever we are working on and be able to provide some help or advise while understanding our work. Time management was not a problem with the team except that we got project for other classes and had to put senior design on hold for a few weeks. As a project manager I feel that I didn’t do a good job. I think I could have done a better job is I had organized myself and set some standards to the way we would communicate our individual work to each other. My partner worked hard and I would definitely be able to work with him again.

Genaro Velasquez:

The project was managed by dividing up the task between both members of the group. As issues came up for each individual task the team member would handle it as be as he could but if he could solve the problem on his own he would ask the other member for assistants. As the project deadline came closer the task that were not finished seemed to blend and both member were then in charged. The schedule for each task was made to so that each task was finished before a little earlier than expected, this way we were rushed to finish the project. When it came to working in teams the I learned that it is best to keep everything in order and to keep constant communication with your partner. Although most of the times were working individually the main problem lied in when we’re trying to combine all of our codes together. I would have been helpful if had set some specific regimen that would have helped in combining our code. Another thing that I learned is that discussing the general goal and specific ones together is very helpful to see where everyone lies and to see what everyone thinks is happening with the project. Ramiro and I spent several hours just discussing the task, which helped us alot when it came writing the output for each individual code. In terms of time management I learned that time management is crucial especially when you are juggling two or three other classes. I learned that setting small goals is essential for time management, is did manage to do it then everything would have been crammed towards the end.

10.2 Packaging and installation issues

We did have to user certain packages for our software to work properly. Some of those packages are Opencv and ccc_win. We also had to use some functions that were provided by Logitech to be able to have the camera pan and tilt. We mainly used all of these packages but had problems joining the different modules together when we were done with each task individually. This was mainly because we were programming in different OS platforms. The final software is suppose to be run off windows using Visual Studios. The installation of these packages was not difficult. We just followed the instructions and were able to use them right away.

10.3 Restrictions, limitations, and constraints

Some of the restrictions on our project were money, size, power, weight, and time.

Money was a problem because we didn't have the budget to buy a better base for the robot or better wheels. The wheels became a problem when we added the laptop to the robot. We had originally set our budget to be 500 dollars but ended up spending about 700 dollars which put both of us in a tight situation financially. If we would have had a bigger budget, I'm pretty sure we could have made the robot a lot better and it would have worked better.

The size shows to be a problem due to the budget problem. Without money we can't really scale the project to the size that we really want it. At the current size the robot is not comfortable for the user to input a room number and take them to the door.

Power becomes a problem once we start adding weight to the robot. The more it has to carry the more power it uses. We ended up being limited to testing due to not having batteries most of the time. We didn't equip ourselves well when testing.

Weight is part of the size and power problem. If the laptop is too heavy for the robot to carry then it will be forced to use most of the power and of course the robot is not big so that plays a big factor in our testing.

Time was a factor because we had other project that also needed our attention and therefore we ended up with sleepless nights. We didn't expect for our classes to be giving us project and had to work around everything to get the results that we got. I think that with more time we could have done a way better job and our robot would be more impressive.

11 * Experiment Design and Test Plan

11.1 * Design of Experiments

Camera: (Ramiro Diaz)

The camera will be tested to make sure that it will detect the door using markers. The output should be a 1 if it detected a door and 0 if the door was not detected in the current frame.

1. Objective is to have door detect a door.
2. The experiment will take place in the Engineering building 2 first floor.
3. Procedures
    a. It will start panning towards where the door should be and detect a door. If the door was detected then the computer should output a message saying so. If not then it should say so as well.
    b. For the previous step to work we need to place markers in the doors and make sure they are at a level where the camera will pick them up.
4. The expected results are to receive the number of doors that are between the starting location and the destinations. That is if there are 4 doors in the path then the camera will have to respond to 4 doors.

Mapping: (Genaro Velasquez)

The Map code will have to generate a path that the camera will need and that the microcontroller will need. If the path is not generated correctly then that is an error and it needs to be taken care of before moving on. The expected result would be having a vector that contains the path for both the camera and the microcontroller.

1. Objective is to have the Map code generate a path.
2. The experiment only needs a room number. It would have good to have a function that will make sure that the path is correct and to check that the camera code got the vector as well as the microcontroller.
3. Procedures:
    a. Run the code and give it a room number
    b. Make a connection to the microcontroller and send the path.
    c. Have the code in the microcontroller running so that the data being transferred is shown on the 7 segment LED.
    d. Next have the camera code output the path in order.
4. The expected results are two vectors that will have the same path. One will go to microcontroller and will be shown on the 7 segment LED. Second is to camera and will be printed to screen or a text file.

Serial communication: (Genaro Velasquez and Ramiro helping debug):

Serial communication will be tested to see if the information is really being sent to the microcontroller. This test is very important and needs the attention of both team members. The test is expected to output the correct information to a 7 segment LED.

1. Objective is to get the data sent through the USB to serial cable displayed in the microcontroller.
2. The experiment will need a basic program running on the microcontroller that will show the data on the 7 segment LED. We will also need to have both team member ready to debug.
3. Procedures:
    a. First make sure the rs232 cable is connected to the laptop and to the microcontroller.
    b. Next is to load the program to the microcontroller and then disconnect.
    c. Next is to run the serial.exe and send a number
4. The expected results should be the display of the number sent using serial.exe on the 7 segment LED.

Sensors (Genaro Velasquez):

This experiment involves the response of the sensors and the decisions it makes to avoid hitting the way and objects. This experiment should expect either a hit or avoidance of a hit.

1. Objective is to test the sensors and the distance at which they respond.
2. The experiment involves the robot itself and the sensors. The tests can be run at any hallway or room.
3. Procedures:
    a. First make sure all the sensors are working properly and that they are connected correctly.
    b. Second load the program to microcontroller and make sure the robot has power for sensors and motor.
    c. Run the test and video record for best debugging.
4. The expected results are avoidance of the walls and objects. Sometimes the sensors get wrong readings so once every 10 tries the robot is expected to hit.

Motors (Genaro Velasquez):

This experiment involves the testing of the wheels and the frequency that will need to be given to each motor to make the wheels move forward or backwards.

1. Objective is to find the frequencies that will make the motors turn and then figure out if the motors do make the robot go forward or backwards and turn.

2. The experiment will need to be trial and error. If the frequency is not good enough then the code will need to be modified to make it better.
3. Procedures:
    a. First find the frequencies.
    b. Second write code for the movement of the robot. (Forward, backwards, right and left)
    c. Next load the program to microcontroller and run the test.
    d. Modify the code as needed and start all over from part c.
4. The expected results are for the robot to move in the direction instructed by the programmer. If the robot does not do as its told then the code need to be modified or the hardware needs to be checked.

Software merging (Ramiro Diaz):

This experiment involves testing the software as we are merging it. Ramiro is in charge of this but Genaro is attentive since we might have to modify code.

1. The objective of this experiment is to successfully merge all the code and have tested everything as we are merging it. The results should be a working software system.
2. This experiment involves all the modules of software and is expected to give us lots of problems when merging them.
3. Procedures:
    a. Fist is to figure out the order of the code. What is going to execute first and what are the problems when merging code from different platforms.
    b. Next is to try to run the map code first and so that a path is generated.
    c. Next add the send serial code and test the code to this point.
    d. Add the camera code and make sure that the camera is panning to the right direction.
    e. Finally connect to the microcontroller and make sure that the top level software is communicating with the low level software.
4. The results show be that everything works out correctly together. The flow of data is smooth and no errors occur.

System Test (Ramiro Diaz and Genaro Velasquez):

This is the test of the entire system working together. Hardware and software are included.

1. The objective is to make sure that the final product does what it was intended to do.
2. This experiment requires the use of the entire first floor of the UCR ENGR2 building. It also requires a controlled environment. Meaning no people to interfere with the experiment and it also requires for video recording at all time.

3. Procedure:
    a. First make sure all connections are where they are suppose to be.
    b. Test sensors one more time
    c. Test camera one more time
    d. Test autonomous navigation
    e. Load the combined code and start the test
4. The results should be that the robot makes it to the room number that is given to it at the beginning of the test. If the robot does not make it then changes need to be made to code or hardware on the spot.

11.2 * Bug Tracking

All the bugs found need to be talked about if they are found when running different modules together so that the individuals in charge of those modules talk it over and find a solution.

Computer Vision:

Ramiro Diaz is in charge of this part of bug tracking. He is to find test his code multiple times until the software does what it's supposed to without any problems. It's also tested with the other programs to see how it reacts to the merge. The expected bugs in this module are related to opening the stream of video from the camera to the computer and panning and resetting correctly. Some errors are expected and are ok for our project. The way Ramiro is to take care of this part is to test and test while going back to the code and making the necessary changes and then testing it again. We don't keep track of bugs since it's a simple testing procedure that doesn't really require any data gathering.

Sensors:

Genaro Velasquez is in charge of this part of bug tracking. Genaro will create a test that will give certain reading to a seven segment LED. This way we can know what state the robot is in. If the robot hits a wall Genaro will quickly check for connections on the sensor and then check the code again to confirm whether it's a hardware problem or software problem. For testing sensors its good to have more than one person watching. Sometimes it's good to video tape the test too. This step is really important since the robot will depend a lot from the sensors.

Mapping:

Mapping is done by both Genaro and Ramiro. Genaro is the main tester for this part. Ramiro is helping with the data structures. This part will most likely have bugs dealing with segfaults and getting the wrong path back. Also the Mapping needs to show the gui like interface for the user to input the room number. That needs to be tested so it shows a small scale of the ENGR2 building.

Software merging:

This part of the testing is done by Ramiro Diaz. Merging code can be tedious and bugs tend to come up very often. Being able to work with Geniro to answer questions is very important at this point. Data is not gathered here. The way this part is debugged is by fixing all the problems that come up and by making changes to the software that we think might be causing the problem.

11.3 * Quality Control

The way we did quality control was different than normal. We stop working on our task to fix a bug that came up that turned out to be a challenge to either one of us. Once we figure out the problem we would then test the system and make sure that it was fully gone. Then we would continue with our own task. Individual bugs that came up sometimes were easily fixed and were not need the attention of all the team members. If the project was at a larger scale then we would have needed a system to log the bugs and get back to them. The main problem we had and were able to both work on was the serial communication and the merging of all the code. Quality Control with us was mostly first come first serve.

11.4 Performance bounds

Some special performance bounds are that the doors need to have a marker that the camera is looking for. With out that marker the camera will not be able to find a door and thus our project would not find the location its looking for. We also assume that the robot always starts at the home position.

11.5 * Identification of critical components

The pats that we saw needed the most attention were serial communication and autonomous navigation. Serial communication is a serious part of our project because if the computer can't communicate with the microcontroller then the robot will be lost and won't know where to go or when it has seen a door. Autonomous navigation is important because it has to be able to avoid objects. If there was no autonomous navigation then the robot would be useless and different ways to get the robot to the destination could be unsafe to the people and to the robot itself. And example would be for the robot to follow a black line and when it got to the door it would stop. Again this is unsafe due to the amount of people that could be walking in the building. The robot needs to be able to avoid them or at least stop so that the people can walk around it.

All parts of the design of the robot are important but as mentioned some are more important than others.

11.6 * Items Not Tested by the Experiments

The only thing that we don't test in the experiment is having the robot return to its home position. This is mainly because if we can get the robot to the door then we could easily get it back to the home position by reversing the directions and by having it read the doors again. Everything else was tested and retested for best performance.

12 * Experimental Results and Test Report

12.1 * Motors

12.1.1 Make motors move

Genaro Velasquez and Ramiro Diaz

1. The operating range for is 140Hz to 350Hz.
2. Expected results were 0 to 400Hz
3. Adjusted PWM on microcontroller

12.1.2 Make motors in a specified direction

Genaro Velasquez and Ramiro Diaz

1. 100Hz-120Hz reverse directions, 220Hz-240Hz no movement, and 240Hz-350Hz forward direction
2. Created a several function that control speeds

12.1.3 Make motors move resulting chassis to turn 90 degrees

Genaro Velasquez and Ramiro Diaz

1. The results were not consistent because the amount the chassis turns is based off of how much power the battery has
2. The expected results were that the motors move the same amount every single time
3. Tried using encoders to adjust the amount of space that the chassis moves

12.1.4 How long will the battery last

Genaro Velasquez and Ramiro Diaz

1. The battery last only five runs with the laptop on top
2. Expected the battery would last longer at least for 10 runs
3. No real actions were taken because there was not enough money to buy an extra battery

12.2 * Sensors

12.2.1 Measure distance from wall

Genaro Velasquez

1. On average the results were correct values but occasionally results outputted were incorrect
2. Genaro expected that the results may fluctuate a bit but never that way off
3. Genaro designed a filter that would throw out the bad results

12.2.2 Detect an intersection

Genaro Velasquez

1. The number outputted from the sensors where high like six thousand centimeters and higher

2. The higher numbers were rounded off to seven thousand and if the left and right sensors had values equal to seven thousand that signified that we have arrived at an intersection

12.2.3 Use code to make avoid and/or go around objects

Genaro Velasquez

1. The motor response is not quick enough to avoid the walls when Jeeves was angled more than thirty degrees
2. Made the motors make harder turns as they got closer to the walls

12.2.4 Determine threshold for hits and misses

Genaro Velasquez

1. About ninety six percent of the values were hits
2. The expected results were less hits, so that outcome was better than expected
3. Added a capacitor for better results

12.3 * Camera Vision

12.3.1 Make C++ recognize the image (pixel method)

Ramiro Diaz

1. The experiment was a success. After changing from using average to concentrating on one or more pixels I got the code to recognize the marker on the doors correctly.
2. The expected results match the experiment results. The door is recognized based on a color target that will be taped to the door for presenting.
3. The experiment result look very good and we can move on to the next part of the project.
4. The only corrective action taken was to increase the range of the values that the pixel could expect.

12.3.2 Binarize images

Ramiro Diaz

1. This experiment seemed to give good results but we also had a different method that we wanted to try. Based on the results of both methods, this experiment and method are being replaced by the pixel method.
2. Expected results for this experiment were favorable but not strong. This is comparing them to the other method that involves pixel level detection.
3. The experiment is a success to a certain point. We are leaving this method and going with something different.
4. We are using a different approach to the problem of detecting a door.

12.3.3 Compare images

Ramiro Diaz

1. This method is part of the binarize of images. Since the results of this method compared to the pixel range method are not good, we have decided to change methods. The results are basically an average of the normalized images. This is done after binarizing them. That’s how we get a threshold and

range that will be used to compare images. This is not a good way to do it since we are only interested in a few pixels.
2. The expected results were a little off the target. I was expecting to get lots of hits since I concentrated more on the marker that I have on the doors. But again comparing this with the pixel method, its obvious that we need to change.
3. Basically we have found out that we need to change methods. Comparing images take longer and it doesn't do as well as the pixel method.
4. The best thing to do is drop this method and quickly change to the pixel method.

12.3.4 Determine optimal percentage that would result in less misses

Ramiro Diaz

1. The experiment results seem to be good ones. The range that gives the more hits has been found and will be soon incorporated to the code.
2. Expected results were matched and are now moving to the implementation of the found data.
3. The experiment showed that concentrating our search within a range of RGB values would give us the most hits for finding a door.
4. All we need to do is increase the range on the code and that should be all need to do.

12.4 * Mapping

12.4.1 Map first floor in C++

Genaro Velasquez

1. The GUI came out to be a bit miss shaped so we adjusted it by retaking the measurements
2. Added new points to the hard coded map and took out the wrong ones

12.4.2 Implement Dijktras

Genaro Velasquez

1. The first two versions of Dijktras came out incorrect because the algorithm looked for the shortest path to all the nodes instead of the destination node.
2. Added an "if" statement that jumped out of loop if the current node equals the destination node

12.4.3 Output all necessary information at each node. Like a distance and direction

Genaro Velasquez

1. This code does not output the distance for each node but it does output the direction to the next node
2. Unfortunately the output of distance was not necessary because the GUI does not stay open long enough

12.5 * Interfacing

12.5.1 User interface

Genaro Velasquez

1. The user interface is part of the mapping; it just interprets the information that happens in the background.

12.5.2 Microcontroller and PC

Genaro Velasquez

1. The values were properly transferred to microcontroller
2. The results were as expected
3. We could have tried making it faster

13 * Conclusion and Future Work

13.1 * Conclusion

Project Jeeves the Robot was a success once we scaled down the requirements. The original project involved having a user friendly interface and speech recognition. But right away we saw that we were not going to be able to accomplish these two requirements and decided to put them on extra. Our project responsibilities were to take a room number from a user and generate the fastest path to get there. Once it found the path it would display to screen and take the user to the destination. While going to the destination if would avoid objects and keep away from the walls. Once it arrived at the destination it would stop. So our project did accomplish all of the objectives. Except that we had a problem with one of the sensors and we had run out of time to debug it. But it didn't affect the project that much.
Our project performed the following:

- Got a room number from the user
- Generated the shortest path to the room number.
- Sent the path to the microcontroller and started the camera
- When the camera recognized one of our markers that represented the door the robot would output that it had found a door to the terminal.
- Once the robot reached the room number it would stop.
- Avoid hitting the walls.

In conclusion our project did do what we had planned. We did end up with a small gui to show the user the map of the building and the path it was going to take. Project was successful but I think it could be better.

Both team members worked hard in this project, but Genaro gave it all when coding the autonomous part of the project. He is a good team member and I hope to work with him in the future.

13.2 Future Work

Jeeves the Robot has potential to being a future senior design project. There are still a lot of functionalities that could be implemented. It could have speech recognition, and of course a better autonomous navigations system. As time goes by more and more technology is coming out and soon we will see guides or assistants like Jeeves on big buildings. There are many ways that this project could be improved. I hope people will find this project interesting and will consider working on it.

Ramiro Diaz
Personally I plan on making the robot better. Since I won't have the pressure of time, I think I can make a few improvements to our current model. One of the improvements that I see would be to have the sensors working a little better. They might be replaced with longer range sensors for better navigation. I would like to add speech recognition to this project and add a little more intelligence since the map is hard coded into the system.

Genaro Velasquez

13.3 Acknowledgement

Be considerate and credit all those who have helped you in this project.
Team Jeeves the Robot would like to thank the following people for their help and support. Without their help our project would have been a failure.

- Northrop Grumman
    - Financial support
- EE Shop and people working there
    - Material, technical advice
- Bourns College of Engineering
    - Usage of the Engineering Unit 2 Building.
- Dr. Ping Liang
    - Advice and ideas
- Hoang Nguyen
    - Support and usage of laptop (big part of project)
- Ankit Patel
    - Programming support
- Abdul Zahid
    - Programming support
- All classmates that help with ideas and cheered the project on.

14 * References

[1] Acroname Robotics, "Sharp GP2D12 Sensor" *acroname.com*. [Online]. Available: http://www.acroname.com/robotics/parts/R48-IR12.html

[2] Dimension Engineering, "Sabertooth 2x10 RC Users Guide" *dimensionengineerng.com*. [Online]. Available: http://www.dimensionengineering.com/datasheets/Sabertooth2x10RC.pdf.

[3] freescale Semiconductor, "MC9S12DP256 SCI Interrupt driven help" *freescale.com*. [Online]. Available: http://forums.freescale.com/freescale/board/message?board.id=16BITCOMM&thread.id=5508

[4] Horstmann, "BigCPP" *horstmann.com*. [Online]. Available: http://horstmann.com/bigcpp.html

[5] Logitech, "How to control pan/tilt/zoom" *quckcamteam.net*. [Online]. Available: http://www.quickcamteam.net/documentation/how-to/how-to-control-pan-tilt-zoom-on-logitech-cameras

[6] Lynxmotion, "4WD2 RC Robot Combo Kit" *lynxmotion.com*. [Online]. Available: http://www.lynxmotion.com/Product.aspx?productID=69&CategoryID=

[7] MC9S12DP256, "Datasheet" *eecs.vanderbilt.edu*. [Online]. Available: http://eecs.vanderbilt.edu/Courses/ee276/Fall2006_downloads/MC9S12DP256.pdf

[8] OpenCV, "Opencv" *opencvlibrary.sourceforge.net*. [Online]. Available: http://opencvlibrary.sourceforge.net/Welcome. [Accessed Feb. 20, 2008].

[9] Wytec Dragon12 Development Board, "Minidragon Board" *www.mecheng.adelaide.edu*. [Online]. Available: http://www.mecheng.adelaide.edu.au/robotics/wpage.php?wpage_id=56

15 Appendices

Presents information that supplements the design specification, including:

Appendix A: Parts List

- Sharp GP2DP12 Sensors
- Laptop
- Logitech Orbit QuickCam
- Lynxmotion 4WD robotic base
- Mini Dragon MC9S12DP256 microcontroller
- 7.2 volt battery
- 9 volt batteries
- Cable
- PCB Boards
- L shaped metal
- Screws

Appendix B: Equipment List

- Wave Functions Generator
- Power supply
- Computer
- Solderer
- Drill gun
- Saw

Appendix C: Software List

- Microsoft Visual Studios 2008
- Cygwin
- Windows Xp OS

16 Code

16.1 Map

16.1.1 main.cpp

```
#include "Mapping.h"
#include "functions.h"
#include "MySerial.cpp"
#include <vector>
#include <string>

using namespace std;

const int x = 100;
int y = 15;
string numTOstr(int num);
void val_send(vector<int> Nodepath)
{
  string send = numTOstr(Nodepath.size()) + "\n";
  printf(send.c_str());
  cwin << Message(Point(x, y), send);
  for(int i = 0; i < Nodepath.size(); i++)
    {
      sleep(1);
      y = 10 + y;
      send = numTOstr(Nodepath[i]) + "\n";
      printf(send.c_str());
      cwin << Message(Point(x, y), send);
    }
}

int location(string cur)
{
  int here = 0;

  if(cur == "Home"){
    here = 1;
  } else if(cur == "136"){
    here = 2;
  } else if(cur == "135"){
    here = 3;
  } else if(cur == "inter1"){
    here = 4;
  } else if(cur == "133"){
    here = 5;
  } else if(cur == "132"){
    here = 6;
  } else if(cur == "inter2"){
    here = 7;
  } else if(cur == "129"){
    here = 8;
```

```
  } else if(cur == "128"){
    here = 9;
  } else if(cur == "127"){
    here = 10;
  } else if(cur == "126"){
    here = 11;
  } else if(cur == "inter3"){
    here = 12;
  } else if(cur == "125"){
    here = 13;
  } else if(cur == "Restrooms"){
    here = 14;
  } else if(cur == "137"){
    here = 15;
  } else if(cur == "inter4"){
    here = 16;
  } else {
    here = 0;
  }

  return here;
}
int ccc_win_main()
{
  cwin.coord(-20,-20, 170,170);
  Mapping floor;
  floor.g_map();
  vector<int> Nodepath;
  int i = 0;

  string place = cwin.get_string("Enter the location you would like to go: ");
  while(place == "" or location(place) == 0) place = cwin.get_string("Enter
the location you would like to go: ");
  vector<int> path = floor.dij(floor.home, place, Nodepath);
  floor.connect(path);
  val_send(Nodepath);

  int tempo = location(place);
  printf("%d", tempo);
  exit(tempo);
  return tempo;
}

string numTOstr(int num)
{
  if(num == 1) return "1";
  else if(num == 2) return "2";
  else if(num == 3) return "3";
  else if(num == 4) return "4";
  else if(num == 5) return "5";
  else if(num == 6) return "6";
  else if(num == 7) return "7";
  else if(num == 8) return "8";
  else if(num == 8) return "9";
```

```
  else return "0";
}
```

16.1.2 Mappin.h

```
#ifndef __MAPPING_H
#define __MAPPING_H

//This declares the Map class
#include <vector>
#include "ccc_win.h"
#include <string>
#include "functions.h"
#include <string>

using namespace std;

class Mapping
{
  //The following are pointers to each room and intersection
 public:
  Node* start;
  Node* home;
  Node* rm136;
  Node* rm135;
  Node* inter1;
  Node* rm133;
  Node* rm132;
  Node* inter2;
  Node* rm129;
  Node* rm128;
  Node* rm127;
  Node* rm126;
  Node* inter3;
  Node* rm125;
  Node* restrooms;
  Node* shop;
  Node* inter4;

  //Creates a map of the first floor by connecting the nodes
  //with certain attributes in the way that makes the node represent
  //the first floor.
  Mapping()
    {
      home = new Node(false,false,"Home",40,180,0,0,0);
      start = home;

      rm136 = new Node(false,true,"136",180,775,0,0,1);
      home->left = rm136;
      rm136->right = home;
      rm135 = new Node(false,true,"135",775,176,0,0,2);
      rm135->right = rm136;
      rm136->left = rm135;
      inter1 = new Node(true,false,"inter1",176,0,114,0,3);
      rm135->left=inter1;
```

```
      inter1->right = rm135;
      rm133 = new Node(false,true,"133",0,0,610,114,4);
      inter1->up = rm133;
      rm133->down = inter1;
      rm132 = new Node(false,true,"132",0,0,148,610,5);
      rm133->up = rm132;
      rm132->down = rm133;
      inter2 = new Node(true,false,"inter2",82,0,0,148,6);
      rm132->up = inter2;
      inter2->down = rm132;
      rm129 = new Node(false,true,"129",340,82,0,0,7);
      inter2->right = rm129;
      rm129->left = inter2;
      rm128 = new Node(false,true,"128",468,340,0,0,8);
      rm129->right = rm128;
      rm128->left = rm129;
      rm127 = new Node(false,true,"127",191,468,0,0,9);
      rm128->right = rm127;
      rm127->left = rm128;
      rm126 = new Node(false,true,"126",152,191,0,0,10);
      rm127->right = rm126;
      rm126->left = rm127;
      inter3 = new Node(true,false,"inter3",0,152,0,146,11);
      rm126->right = inter3;
      inter3->left = rm126;
      rm125 = new Node(false,true,"125",0,0,146,232,12);
      inter3->down= rm125;
      rm125->up = inter3;
      restrooms = new Node(false,false,"Restrooms",0,0,232,385,13);
      rm125->down = restrooms;
      restrooms->up = rm125;
      shop = new Node(false,true,"137",0,0,385,154,14);
      restrooms->down = shop;
      shop->up = restrooms;
      inter4 = new Node(true,false,"inter4",0,80,154,0,15);
      shop->down = inter4;
      inter4->up = shop;
      inter4->left = home;
      home->right = inter4;
    };
  //declares the function that will be used
  vector<int> dij(Node* cur, string last, vector<int>& direction);
  Node* min_N(vector<Node*>& Q);
  void relax_n(Node* u, vector<Node*>& Q, vector<Node*>& S);
  bool in_arr(vector<Node*> S, Node* u);
  void g_map();
  void connect(vector<int> path);
  char orientation(Node* cur, Node* prev);
};

//Calculates the path that needs to be taken to arrive at destination
vector<int> Mapping::dij(Node* cur, string dest, vector<int>& dir)
{
  //Sets all the information to initial settings
```

```
  vector<Node*> Nodepath;
  vector<int> direction;
  home->Pred = NULL;
  rm136->Pred = NULL;
  rm135->Pred = NULL;
  inter1->Pred = NULL;
  rm133->Pred = NULL;
  rm132->Pred = NULL;
  inter2->Pred = NULL;
  rm129->Pred = NULL;
  rm128->Pred = NULL;
  rm127->Pred = NULL;
  rm126->Pred = NULL;
  inter3->Pred = NULL;
  rm125->Pred = NULL;
  restrooms->Pred = NULL;
  shop->Pred = NULL;
  inter4->Pred = NULL;

//Sets all the information to initial settings
  home->dist = 20000;
  rm136->dist = 20000;
  rm135->dist = 20000;
  inter1->dist = 20000;
  rm133->dist = 20000;
  rm132->dist = 20000;
  inter2->dist = 20000;
  rm129->dist = 20000;
  rm128->dist = 20000;
  rm127->dist = 20000;
  rm126->dist = 20000;
  inter3->dist = 20000;
  rm125->dist = 20000;
  restrooms->dist = 20000;
  shop->dist = 20000;
  inter4->dist = 20000;

//Sets all the information to initial settings
  home->visit = false;
  rm136->visit = false;
  rm135->visit = false;
  inter1->visit = false;
  rm133->visit = false;
  rm132->visit = false;
  inter2->visit = false;
  rm129->visit = false;
  rm128->visit = false;
  rm127->visit = false;
  rm126->visit = false;
  inter3->visit = false;
  rm125->visit = false;
  restrooms->visit = false;
  shop->visit = false;
  inter4->visit = false;
```

```
cur->dist = 0;
cur->visit = true;
vector<int> ret;
vector<Node*> Q;
vector<Node*> S;

Q.push_back(cur);
cur->dist = 0;

Node* u;
//while Q is not empty
while(Q.size() > 0)
  {
    //u = extract-minimum(Q)
    u = min_N(Q);
    //add u to S
    S.push_back(u);
    if(u->room == dest)
       {
         break;
       }
    // relax-neighbors(u)
    relax_n(u, Q, S);
  }

while(u != NULL)
  {
    ret.push_back(u->loc);
    //pushing node to Nodepath vector
    Nodepath.push_back(u);
    u = u->Pred;
  }
vector<Node*> temp;

//reversing the Nodepath vector
for(int i = Nodepath.size()-1; i >=0; i--)
  {
    temp.push_back(Nodepath[i]);
  }
Nodepath = temp;

char direct;
char pre = '0';

//For each NODE it checks the path to the next NODE
//and it will output a number depending on the direction
//1 = forward, 2 = left, 3 = right.
for(int i = 0; i < Nodepath.size(); i++)
  {
    if(i != Nodepath.size() - 1)
    {
      direct = orientation(Nodepath[i], Nodepath[i + 1]);
```

```
        if(pre != '0'){
            if(direct == pre){
              direction.push_back(1);
            }
            else if(direct == 'n' && pre == 'e'){
              direction.push_back(2);
            }
            else if(direct == 's' && pre == 'e'){
              direction.push_back(3);
            }
            else if(direct == 'w' && pre == 'n'){
              direction.push_back(2);
            }
            else if(direct == 'e' && pre == 'n'){
              direction.push_back(3);
            }
            else if(direct == 'n' && pre == 'w'){
              direction.push_back(3);
            }
            else if(direct == 's' && pre == 'w'){
              direction.push_back(2);
            }
            else if(direct == 'w' && pre == 's'){
              direction.push_back(3);
            }
            else if(direct == 'e' && pre == 's'){
              direction.push_back(2);
            }
          }
        else{
            if(direct == 'w'){
              direction.push_back(2);
            }
            else{
              direction.push_back(3);
            }
          }
      }
      pre = direct;
    }
  dir = direction;
  return ret;
}

//checks if the prvious node is north, west, south or east
//compared to the current node.
char Mapping::orientation(Node* cur, Node* prev)
{
  if(cur->right == prev){
      return 'e';
    }
  else if(cur->left == prev){
      return 'w';
    }
```

```
  else if(cur->up == prev){
      return 'n';
    }
  else{
      return 's';
    }
}

void Mapping::relax_n(Node* cur, vector<Node*>& Q, vector<Node*>& S)
{
//for each vertex v adjacent to u, v not in S
// a shorter distance exists
      if(cur->right != NULL and !(in_arr(S, cur->right)))
      {
        if(cur->right->dist > cur->rdist + cur->dist)
          {
            cur->right->dist = cur->rdist + cur->dist;
            cur->right->Pred = cur;
            Q.push_back(cur->right);
          }
      }
      if(cur->left != NULL and !(in_arr(S, cur->left)))
      {
        if(cur->left->dist > cur->ldist + cur->dist)
          {
            cur->left->dist = cur->ldist + cur->dist;
            cur->left->Pred = cur;
            Q.push_back(cur->left);
          }
      }
      if(cur->up != NULL and !(in_arr(S, cur->up)))
      {
        if(cur->up->dist > cur->updist + cur->dist)
          {
            cur->up->dist = cur->updist + cur->dist;
            cur->up->Pred = cur;
            Q.push_back(cur->up);
          }
      }
      if(cur->down != NULL and !(in_arr(S, cur->down)))
      {
        if(cur->down->dist > cur->ddist + cur->dist)
          {
            cur->down->dist = cur->ddist + cur->dist;
            cur->down->Pred = cur;
            Q.push_back(cur->down);
          }
      }
}
Node* Mapping::min_N(vector<Node*>& Q)
{
  Node* ret;
  double temp = 20000;
  int pos = 0;
```

```
  for(unsigned int i = 0; i < Q.size(); i++)
    {
      if(Q[i]->dist <= temp)
      {
        temp = Q[i]->dist;
        ret = Q[i];
        pos = i;
      }
    }
  vector<Node*>::iterator iter;
  iter = Q.begin();
  Q.erase(iter + pos);
  return ret;
}

//This function checks wether or not the Node pointer
//is in the vector of pointers.
bool Mapping::in_arr(vector<Node*> S, Node* u)
{
  for(unsigned int i = 0; i < S.size(); i++)
    {
      if(u->loc == S[i]->loc)
          {
            return true;
          }
    }
  return false;
}

//Outputs the map of the first floor to the screen
void Mapping::g_map()
{
  Point arrA[82] = {
    Point (0, 15.9),
    Point (0, 0),
    Point (7.3, 0),
    Point (7.3, 18.7),
    Point (10.7, 18.7),
    Point (10.7, 17.1),
    Point (19.81, 17.1),
    Point (19.81, 18.7),
    Point (22.51, 18.7),
    Point (22.51, 17.1), ///10
    Point (33.01, 17.1),
    Point (33.01, 18.7),
    Point (35.11, 18.7),
    Point (35.11, 17.1),
    Point (48.21, 17.1),
    Point (48.21, 18.7),
    Point (50.31, 18.7),
    Point (50.31, 17.1),
    Point (60.81, 17.1),
    Point (60.81, 18.7), //20
```

```
Point (62.81, 18.7),
Point (62.81, 17.1),
Point (71.81, 17.1),
Point (71.81, 18.7),
Point (75.11, 18.7),
Point (75.11, 2.9),
Point (83.91, 2.9),
Point (83.91, 13.1),
Point (84.91, 13.1),
Point (84.91, 16.3), //30
Point (83.91, 16.3),
Point (83.91, 19.1),
Point (88.41, 19.1),
Point (88.41, 25.6),
Point (81.61, 25.6),
Point (81.61, 42.6),
Point (83.01, 42.6),
Point (83.01, 124.6),
Point (87.51, 124.6),
Point (87.51, 132.4),//40
Point (81.71, 132.4),
Point (81.71, 165),
Point (73.69, 165),
Point (73.69, 131.6),
Point (71.39, 131.6),
Point (71.39, 133),
Point (63.26, 133),
Point (63.26, 131.6),
Point (61.26, 131.6),
Point (61.26, 133), //50
Point (50.86, 133),
Point (50.86, 131.6),
Point (48.76, 131.6),
Point (48.76, 133),
Point (35.66, 133),
Point (35.66, 131.6),
Point (33.56, 131.6),
Point (33.56, 133),
Point (23.16, 133),
Point (23.16, 131.6),// 60
Point (21.06, 131.6),
Point (21.06, 133),
Point (13.93, 133),
Point (13.93, 131.6),
Point (8.5, 131.6),
Point (8.5, 138.6),
Point (2.13, 138.6),
Point (2.13, 137.2),
Point (-0.53, 137.2),
Point (-0.53, 130.6),//70
Point (-2.53, 130.6),
Point (-2.53, 124.7),
Point (1.53, 123.7),
Point (1.53, 84.8),
```

```
    Point (0.42, 84.8),
    Point (0.42, 71.4),
    Point (1.53, 71.4),
    Point (1.53, 32.9),
    Point (0.42, 32.9),
    Point (0.42, 26.78),//80
    Point (1.53, 26.78),
    Point (1.53, 23.07)};

  Point arrB[33] = {Point (8.5, 26.1),
                Point (20.3, 26.1),
                Point (20.3, 27.1),
                Point (27.9, 27.1),
                Point (27.9, 26.1),
                Point (46, 26.1),
                Point (46, 27.1),
                Point (53.6, 27.1),
                Point (53.6, 26.1),
                Point (74.5, 26.1),//10
                Point (74.5, 42.9),
                Point (73.39, 42.9),
                Point (73.39, 51.9),
                Point (74.5, 51.9),
                Point (74.5, 107.6),
                Point (72.5, 107.6),
                Point (72.5, 117.6),
                Point (74.5, 117.6),
                Point (74.5, 124.6),
                Point (8.5, 124.6),//20
                Point (8.5, 122.4),
                Point (10.1, 122.4),
                Point (10.1, 113.2),
                Point (8.5, 113.2),
                Point (8.5, 46.2),
                Point (9.9, 46.2),
                Point (9.9, 36.9),
                Point (8.5, 36.9),
                Point (8.5, 34.8),
                Point (10.3, 34.8),//30
                Point (10.3, 28.7),
                Point (8.5, 28.7),
                Point (8.5, 26.1)};

  for(int i = 0; i < 81; i++)
    {
      cwin << Line(arrA[i], arrA[i + 1]);
      if(i < 32)
          {
            cwin << Line(arrB[i], arrB[i + 1]);
          }
    }
  cwin << Message (Point (1, -4), "Exit");
  cwin << Message (Point (11.7, 13.1), "125");
  cwin << Message (Point (62.81, 13.1), "137");
```

```
  cwin << Message (Point (78.11, -1.9), "Exit");
  cwin << Message (Point (75.69, 165), "Exit");
  cwin << Message (Point (65.26, 134), "133");
  cwin << Message (Point (15.93, 133), "132");
  cwin << Message (Point (5.23, 139.6), "Exit");
  cwin << Message (Point (-7.42, 74.8), "128");
  cwin << Message (Point (-7.82, 26.7), "126");
  cwin << Message (Point (20.3, 27.1), "Restrooms");
  cwin << Message (Point (64.5, 45.0), "136");
  cwin << Message (Point (64.5, 110.5), "135");
  cwin << Message (Point (10.5, 115.9), "129");
  cwin << Message (Point (10.5, 39.9), "127");
}

//This function outputs the path  that the dij algorithm calculates
//It uses defined points to outputs circles and uses lines to connect them
void Mapping::connect(vector<int> path)
{
    Point ar[] = {Point (79.00, 30.77),
    Point (79.00, 47.25),
    Point (79.00, 112.00),
    Point (79.00, 128.50),
    Point (68.00, 128.50),
    Point (18.00, 128.50),
    Point (5.50, 128.50),
    Point (5.50, 118.30),//10
    Point (5.50, 77.80),
    Point (5.50, 42.30),
    Point (5.50, 30.40),
    Point (5.50, 22.00),
    Point (15.10, 22.00),
    Point (24.17, 22.00),
    Point (67.10, 22.00),
    Point (79.00, 22.00)
      };//20

    for(unsigned int i = 0; i < path.size(); i++)
      {
           if(i < path.size() -1)
             {
               cwin << Line (ar[path[i]], ar[path[i + 1]]);
             }
      for(double j = 2; j > 0; j = j - 0.1)
             {
               cwin << Circle (ar[path[i]], j);
             }
      }
}


#endif
```

16.1.3 functions.h

```
#ifndef __FUNCTIONS_H
```

```
#define __FUNCTIONS_H

#include<vector>
#include<string>
//***************

using namespace std;

//This function declares a new class
class Node
{
  //These are public variables
 public:
  bool isinter;
  bool isdoor;
  bool visit;
  string room;
  double rdist;
  double ldist;
  double updist;
  double ddist;
  int loc;
  double dist;

  Node* left;
  Node* right;
  Node* up;
  Node* down;
  Node* Pred;

  //This defines the information that each node contains.
  Node()
    {
      isinter = false;
      isdoor = false;
      room = "";
      rdist = 0;
      ldist = 0;
      updist = 0;
      ddist = 0;
      left = NULL;
      right = NULL;
      up = NULL;
      down = NULL;

      loc = 0;
      dist = -1;
      Pred = NULL;
    };
  //Easy way of declaring a node.
  Node(bool inter,bool door,string room, double rr,
       double ll,double uu, double dd, int lock)
    {
```

```
        isinter = inter;isdoor = door;
        rdist = rr; ldist = ll;
        updist = uu; ddist = dd;
        this->room = room;


        left = NULL;
        right = NULL;
        up = NULL;
        down = NULL;
        loc = lock;
    };

};

#endif
```

16.1.4 Makefile

```
CXX=g++

CXXFLAGS= -D CCC_MSW

LIBS= /usr/local/lib/cccfiles/ccc_msw.cpp /usr/local/lib/cccfiles/ccc_shap.cpp
-luser32 -lgdi32

LIBDIRS=   -I   /usr/include   -I   /usr/include/w32api   -I   /usr/local/lib   -I
/usr/local/lib/cccfiles


OBJ = main.o

all: $(OBJ)

      $(CXX) $(CXXFLAGS) $(LIBDIRS) -o a.exe $(OBJ) $(LIBS)

main.o: functions.h Mapping.h MySerial.cpp main.cpp

      $(CXX) $(CXXFLAGS) $(LIBDIRS) -c main.cpp

clean:

      rm -rf *~ a.exe *.o
```

16.2 Serial

16.2.1 main.cpp

```
#include <iostream>
#include "MySerial.cpp"
using namespace std;

int main(){
      InitSerial();
return 0;
}
```

16.2.2 MySerial.h

```
//#define PORT "/dev/ttyUSB0"
#define PORT "COM2"
#define BUAD B9600

//#include <iostream>
#include <signal.h>
#include <termios.h>
#include <unistd.h>
#include <fcntl.h>

#include <string>
using namespace std;
string Buffer;
int fd_com;
struct termios NewTTY, OldTTY;
struct sigaction saio;

#ifdef SERIAL
bool again = true;
void prompt();
#endif

void com_handler(int status);
void SendSerial(const char *s);
int InitSerial();

#ifdef SERIAL
int main(){
#else
int InitSerial(){
#endif

	fd_com = open(PORT, O_RDWR | O_NOCTTY | O_NONBLOCK);
	if(fd_com < 0){
		printf("\nUnable to open Serial PORT!\n");
		return -1;
	}

	tcgetattr(fd_com, &OldTTY); // save old config
	bzero(&NewTTY, sizeof(NewTTY));

	NewTTY.c_cflag = BUAD | CS8 | CLOCAL | CREAD;
	NewTTY.c_iflag = IGNPAR;
	NewTTY.c_oflag = 0;
	NewTTY.c_lflag = 0;
	NewTTY.c_cc[VTIME] = 0;
	NewTTY.c_cc[VMIN] = 1;

	tcflush(fd_com, TCIFLUSH);
	tcsetattr(fd_com, TCSANOW, &NewTTY);
	fcntl(fd_com, F_SETOWN, getpid());
	fcntl(fd_com, F_SETFL, FASYNC);
```

```
	saio.sa_handler = com_handler;
	sigemptyset(&saio.sa_mask);
	saio.sa_flags = 0;
	//saio.sa_restorer = NULL;
	sigaction(SIGIO, &saio, NULL);

#ifdef SERIAL
	//cout << ":::SERIAL MODE:::" << endl;
	int prompt_wait = 0;
	while(1){
		prompt();
		if(Buffer.size() > 1){
			string cmd = Buffer.substr(0,1);
			Buffer.erase(0,1);
			cout << cmd;
		} else {
			prompt_wait++;
			if(prompt_wait > 30000){
				prompt_wait = 0;
				again = true;
			}
		}
	}
#endif /* SERIAL */
	return 1;
}

#ifdef SERIAL
void prompt(){
	if(again){
	  string inp;
	  //cout << "Send: ";
	  //cin >> inp;
		SendSerial(inp.c_str());
		if(inp == "P" || inp == "F")
			again = false;
	}
}
#endif

void com_handler(int status){
	char tmp[256*2];
	int len = read(fd_com, tmp, sizeof(tmp));
	tmp[len] = 0;
	Buffer += tmp ;
}

void SendSerial(const char *s){
	int err;
	if( (err = write(fd_com, s, strlen(s) ) ) == -1 );
	  //cerr << "Error writing to Serial Port." << endl;
}
```

16.3 Microcontroller

16.3.1 main.c

```
#include <mc9s12dp256.h>                                  /* contains register
definitions */
                                                          /* for the 9S12DP256
*/
static volatile unsigned short rtiCnt;                    /* counts how many
times the RTI*/
static volatile unsigned short wheel;
unsigned int per = 177;//period
unsigned int turn = 0; //# of turns wheel
int tempy = 0;
int count = 0; //keeps track of current position


//enc_right= ecoder right and ecoder left
unsigned long i, enc_right, enc_left;

//sensor variables hold last measured distances
unsigned int leftval, leftval1, rightval, upval, downval, t_turn, here;

//current place from incoming info
int cur_place = 0;

//loc_val = number of directions c=holds current direction
char loc_val, c;
char array[10]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x67};
//holds the path of directions
char direct[50];
//distingush from incoming value and next value
int bool_val = 1;
//---------------------------------------------------------
/*Sensor reading on the bot*/
void IR_Read(void);
void SCI0InterruptHandler(void);
//---------------------------------------------------------
//---------------------------------------------------------
//filtering readings

//global var
int var1 = 0;
int var2 = 0;
int var3 = 0;
int var4 = 0;
int var5 = 0;

//below 50...
int fcnt(int reading)
{
  int temp = 0;
  var5 = var4;
  var4 = var3;
```

```
  var3 = var2;
  var2 = var1;
  var1 = reading;

  if(var1 < 50) temp ++;
  if(var2 < 50) temp ++;
  if(var3 < 50) temp ++;
  if(var4 < 50) temp ++;
  if(var5 < 50) temp ++;

  if(temp >= 3)
  {
    if(var1< 50) return var1;
    if(var2< 50) return var2;
    if(var3< 50) return var3;
  }
  return var1;
}
//---------------------------------------------------------
//filtering readings

//global var
int var11 = 0;
int var21 = 0;
int var31 = 0;
int var41 = 0;
int var51 = 0;

//below 50...
int fcnt1(int reading)
{
  int temp = 0;
  var51 = var41;
  var41 = var31;
  var31 = var21;
  var21 = var11;
  var11 = reading;

  if(var11 < 50) temp ++;
  if(var21 < 50) temp ++;
  if(var31 < 50) temp ++;
  if(var41 < 50) temp ++;
  if(var51 < 50) temp ++;

  if(temp >= 3)
  {
    if(var11< 50) return var11;
    if(var21< 50) return var21;
    if(var31< 50) return var31;
  }
  return var11;
}
//---------------------------------------------------------
//filtering readings
```

```
//global var
int var12 = 0;
int var22 = 0;
int var32 = 0;
int var42 = 0;
int var52 = 0;

//below 50...
int fcnt2(int reading)
{
  int temp = 0;
  var52 = var42;
  var42 = var32;
  var32 = var22;
  var22 = var12;
  var12 = reading;

  if(var12 < 50) temp ++;
  if(var22 < 50) temp ++;
  if(var32 < 50) temp ++;
  if(var42 < 50) temp ++;
  if(var52 < 50) temp ++;

  if(temp >= 3)
  {
    if(var12< 50) return var12;
    if(var22< 50) return var22;
    if(var32< 50) return var32;
  }
  return var12;
}

//global var
int var14 = 0;
int var24 = 0;
int var34 = 0;
int var44 = 0;
int var54 = 0;

//below 50...
int fcnt4(int reading)
{
  int temp = 0;
  var54 = var44;
  var44 = var34;
  var34 = var24;
  var24 = var14;
  var14 = reading;

  if(var14 < 50) temp ++;
  if(var24 < 50) temp ++;
  if(var34 < 50) temp ++;
  if(var44 < 50) temp ++;
```

```
  if(var54 < 50) temp ++;

  if(temp >= 3)
  {
    if(var14< 50) return var14;
    if(var24< 50) return var24;
    if(var34< 50) return var34;
  }
  return var14;
}

//moving forward controlling wheels
void forward() {
 //left wheel
 PWMPER0 = 200;
 //right wheel
 PWMPER2 = 194;
 //duty cycles
 PWMDTY0 = 200 * 0.5;
 //duty cycles
 PWMDTY2 = 194 * 0.5;
}

void forward_r() {
 PWMPER0 = 200;
 PWMPER2 = 200;
 PWMDTY0 = 200 * 0.5;
 PWMDTY2 = 200 * 0.5;
}

void forward_r2() {
 PWMPER0 = 184;
 PWMPER2 = 200;
 PWMDTY0 = 184 * 0.5;
 PWMDTY2 = 200 * 0.5;
}

void forward_l2() {
 PWMPER0 = 200;
 PWMPER2 = 184;
 PWMDTY0 = 200 * 0.5;
 PWMDTY2 = 184 * 0.5;
}

void forward_l() {
 PWMPER0 = 200;
 PWMPER2 = 188;
 PWMDTY0 = 200 * 0.5;
 PWMDTY2 = 188 * 0.5;
}


void backward() {
 PWMPER0 = 100;
```

```
 PWMPER2 = 100;
 PWMDTY0 = 100 * 0.5;
 PWMDTY2 = 100 * 0.5;
}

void right() {
 PWMPER0 = 100;
 PWMPER2 = 200;
 PWMDTY0 = 100 * 0.5;
 PWMDTY2 = 200 * 0.5;
 wheel = 0;
 while(wheel < 32);
}

void right1() {
 PWMPER0 = 100;
 PWMPER2 = 200;
 PWMDTY0 = 100 * 0.5;
 PWMDTY2 = 200 * 0.5;
 wheel = 0;
 while(wheel < 16);
}

void left(){

 PWMPER0 = 200;
 PWMPER2 = 100;
 PWMDTY0 = 200 * 0.5;
 PWMDTY2 = 100 * 0.5;
 wheel = 0;
 while(wheel < 32);
}

void stop(){
 PWMPER0 = 150;
 PWMPER2 = 150;
 PWMDTY0 = 150 * 0.5;
 PWMDTY2 = 150 * 0.5;
 }
typedef enum{start, stop1, intersect, automa, none, avoid, pre_s, pre_start}
Statetype;
//current state
Statetype curst;
//previous state
Statetype prev;
//--------------------------------------------------------

/****************************************************************************
**************/
                                        //Main Driver
/****************************************************************************
**************/
void main(void){
  //real time counter.
```

```
  rtiCnt = 0;                                        /* set counter to 0
*/
  DDRH = 0x7f;                                       /* set porth pins 0-6
too utputs*/
  PTH = 0x00;                                       /* turn off 7 segment
display   */
  RTICTL = 0x70;                                     /* 1/(16MHz/2^16) =
4.096 ms    */
                                                     /* cycle time for RTI
*/
  CRGINT |= 0x80;                                    /* enable the interrupt
for RTI */                                         /* enable the interrupts
*/
  asm("cli");
      PORTE =0x02; /*IRQ PIN PE1 PULL HIGH*/
  INTCR = 0xc0;
  PTH = array[6];
//---------------------------------------------
  //pwm enable
  PWME = 0x05;
  //polling
  PWMPOL = 0x05;
  PWMCLK = 0x05;  // set to use SA
  PWMPRCLK = 0x77;// Set prescale to 128
  PWMSCLA = 0x02;  // set to lowest possible
  PWMSCLB = 0x02;  //sets clock b to lowest possible
  PWMPER0 = 200;
      PWMDTY0 = 200 * 0.5;
      PWMPER2 = 200;
      PWMDTY2 = 200 * 0.5;

      enc_right = 0;
      enc_left = 0;
//---------------------------------------------
  curst = pre_s;//sets the current state

  //run forever.

  while(1) {
    //will switch based on the curst value
    switch(curst) {
//-----------------------------------------------------------------------------
-----------
      case pre_s:
        prev = pre_s;
        stop();
        //PTH = array[0];

        break;
//-----------------------------------------------------------------------------
-----------
      case pre_start:
      prev = pre_start;
      //PTH = array[1];
```

```
        if(direct[0] == 2){
          //left();
          PWMPER0 = 200;
          PWMPER2 = 100;
          PWMDTY0 = 200 * 0.5;
          PWMDTY2 = 100 * 0.5;
          while(upval != 7000);
          stop();
          wheel = 0;
          while(wheel < 300);
          curst = start;
          //PTH = array[9];
        } else if(direct[0] == 3){
          PWMPER0 = 100;
          PWMPER2 = 200;
          PWMDTY0 = 100 * 0.5;
          PWMDTY2 = 200 * 0.5;
          while(upval != 7000);
          stop();
          wheel = 0;
          while(wheel < 300);
          //PTH = array[8];
          curst = start;
        } else {
          stop();
          //PTH = array[7];
          curst = none;
        }
      count = 2;
      break;
//------------------------------------------------------------------------------
-----------
      case start:
        prev = start;//keep track of previous state
        //PTH = array[2];
        PTH = array[count];
        //if the reading is less than 55 cm do following
        if(leftval < 65){
          forward_l2();
        } else if (rightval < 65){
          forward_r2();
        } else if(leftval < 80) {
          forward_l();
        } else if(rightval < 80){
          forward_r();
        } else if(leftval == leftval1){
          forward();
        } else if(leftval == 7000 && rightval == 7000 && direct[count /2]  !=
1){
           curst = intersect;
        } else if(leftval == 7000){
          forward_r();
        } else if(rightval == 7000){
          forward_l();
```

```
        } else {
          forward();
        }

        if(count / 2 == loc_val + 1) {
         curst = none;
        }

        break;
//-----------------------------------------------------------------------------
-----------
      //if it senses that its going to hit then it stops
      case stop1:
          if(rightval < leftval){
             PWMPER0 = 200;
             PWMPER2 = 100;
             PWMDTY0 = 200 * 0.5;
             PWMDTY2 = 100 * 0.5;
          } else {
             PWMPER0 = 100;
             PWMPER2 = 200;
             PWMDTY0 = 100 * 0.5;
             PWMDTY2 = 200 * 0.5;
          }
          curst = start;
          prev = stop1;
        break;
//-----------------------------------------------------------------------------
-----------
      case intersect:
      prev = intersect;
      //PTH = array[4];
      wheel = 0;
      while(wheel < 100);

      stop();
      wheel = 0;
      while(wheel < 40);

      if(direct[count / 2] == 2) {
         left();
      } else if(direct[count / 2] == 3){
         right();
      }

      count = count + 2;
      stop();
      wheel = 0;
      while(wheel < 40);

        curst = start;
        break;
//-----------------------------------------------------------------------------
-----------
```

```
      case avoid:
        prev = avoid;
        //PTH = array[5];
        if(prev != none) {
          forward();
        } else {
          backward();
        }

        if((leftval == 7000) && (leftval1 == 7000)) {
          right1();
          curst = automa;
        } else {
          curst = avoid;
        }

        break;
//-----------------------------------------------------------------------------
-----------
      case none:
        stop();
        PTH = array[count];
        //bool_val = 1;
        //curst = pre_s;
        //prev = none;
        break;
//-----------------------------------------------------------------------------
-----------
    }
  }
 };

__interrupt void RealTimeInterrupt(void)
{
  rtiCnt++;
  wheel++;                                              //increment the counter
  IR_Read();

  if(upval < 60 && curst == start) {
   curst = stop1;
  }
//-----------------------------------------------
  if(bool_val == 1 )
   {
      while(SCI0SR1_RDRF == 0 && SCI0SR1_OR == 0);
      loc_val = SCI0SR1;
      loc_val = SCI0DRL - 48;
      bool_val = 0;
      PTH = array[loc_val];
      cur_place = 0;
   }
  else if(loc_val > cur_place)
   {
      while(SCI0SR1_RDRF == 0 && SCI0SR1_OR == 0);
```

```
      direct[cur_place] = SCI0SR1;
      direct[cur_place] = SCI0DRL - 48;

      PTH = array[direct[cur_place]];

      cur_place++;
      if(cur_place == loc_val){
        curst = pre_start;
        count = 0;
      }
  }
//-------------------------------------------------
  CRGFLG = 0x80;
}

__interrupt void IRQInterrupt(void)
{
   count++;                                   /* toggle 7 segment display
*/
} /*** end of IRQInterrupt ***/

 void IR_Read()
{
  //sets the environment on mc
  ATD1CTL2=0x80;//enables a/d
  ATD1CTL3=0x48;//#of ports to read
  for (i=0;i<15000;i++);//wait
  ATD1CTL4=0x00;//tells which register to start
  ATD1CTL5=0xb0;//tells the mc to do multiple ports 10bit resolution

  //wait for register to hold some value from the sensors
  while((ATD1STAT0_SCF&0x01)==0);

  //register one which corresponds to front sensor
  if(ATD1DR0 > 9) {
  //equation to normalize reading (cm)
  upval = fcnt((6787 / (ATD1DR0 - 3)) - 4);
  } else {
  //else value is too low
  upval = 7000;
  }

  if(ATD1DR1 > 9) {
  rightval = fcnt1((6787 / (ATD1DR1 - 3)) - 4);
  } else {
  rightval = 7000;
  }

  if(ATD1DR2 > 9) {
  leftval = fcnt2((6787 / (ATD1DR2 - 3)) - 4);
  } else {
  leftval = 7000;
  }
```

```
  if(ATD1DR7 > 9) {
  here = (6787 / (ATD1DR7 - 3)) - 4;
  } else {
  here = 7000;
  }

  if(ATD1DR4 > 9) {
  leftval1 = fcnt4((6787 / (ATD1DR4 - 3)) - 4);
  } else {
  leftval1 = 7000;
  }

  if(ATD1DR5 > 9) {
  downval = (6787 / (ATD1DR5 - 3)) - 4;
  } else {
  downval = 7000;
  }
}

/************************************          end          of          main.c
*************************************/
```

16.3.2 vectors.c

```
/*****************************************************************************
***********
* Type definitions
******************************************************************************
**********/
typedef void (*tIsrFunc)(void);                                    /* ISR
function type */


/*****************************************************************************
***********
* Function prototypes
******************************************************************************
**********/
extern __interrupt void RealTimeInterrupt(void);
//extern __interrupt void sci0ISR(void);
//extern __interrupt void PortHInterrupt(void);
extern __interrupt void IRQInterrupt(void);

/*****************************************************************************
***********
* Global constants
******************************************************************************
**********/
const tIsrFunc _vectab[] @0x3e00 =
{
  (tIsrFunc)0,                                          /* Reserved 0xFF80
*/
  (tIsrFunc)0,                                          /* Reserved 0xFF82
*/
```

```
  (tIsrFunc)0,                                          /* Reserved 0xFF84
*/
  (tIsrFunc)0,                                          /* Reserved 0xFF86
*/
  (tIsrFunc)0,                                          /* Reserved 0xFF88
*/
  (tIsrFunc)0,                                          /* Reserved 0xFF8A
*/
  (tIsrFunc)0,                                          /* PWM Emergency
Shutdown 0xFF8C */
  (tIsrFunc)0,                                          /* PortP Interrupt
0xFF8E        */
  (tIsrFunc)0,                                          /* MSCAN4 Transmit
0xFF90        */
  (tIsrFunc)0,                                          /* MSCAN4 Receive 0xFF92
*/
  (tIsrFunc)0,                                          /* MSCAN4 Errors 0xFF94
*/
  (tIsrFunc)0,                                          /* MSCAN4 WakeUp 0xFF96
*/
  (tIsrFunc)0,                                          /* MSCAN3 Transmit
0xFF98        */
  (tIsrFunc)0,                                          /* MSCAN3 Receive 0xFF9A
*/
  (tIsrFunc)0,                                          /* MSCAN3 Errors 0xFF9C
*/
  (tIsrFunc)0,                                          /* MSCAN3 WakeUp 0xFF9E
*/
  (tIsrFunc)0,                                          /* MSCAN2 Transmit
0xFFA0        */
  (tIsrFunc)0,                                          /* MSCAN2 Receive 0xFFA2
*/
  (tIsrFunc)0,                                          /* MSCAN2 Errors 0xFFA4
*/
  (tIsrFunc)0,                                          /* MSCAN2 WakeUp 0xFFA6
*/
  (tIsrFunc)0,                                          /* MSCAN1 Transmit
0xFFA8        */
  (tIsrFunc)0,                                          /* MSCAN1 Receive 0xFFAA
*/
  (tIsrFunc)0,                                          /* MSCAN1 Errors 0xFFAC
*/
  (tIsrFunc)0,                                          /* MSCAN1 WakeUp 0xFFAE
*/
  (tIsrFunc)0,                                          /* MSCAN0 Transmit
0xFFB0        */
  (tIsrFunc)0,                                          /* MSCAN0 Receive 0xFFB2
*/
  (tIsrFunc)0,                                          /* MSCAN0 Errors 0xFFB4
*/
  (tIsrFunc)0,                                          /* MSCAN0 WakeUp 0xFFB6
*/
  (tIsrFunc)0,                                          /* Flash 0xFFB8
*/
```

```
  (tIsrFunc)0,                                                 /* Eeprom WakeUp 0xFFBA
*/
  (tIsrFunc)0,                                                 /* SPI2  0xFFBC
*/
  (tIsrFunc)0,                                                 /* SPI1  0xFFBE
*/
  (tIsrFunc)0,                                                 /* IIC Bus 0xFFC0
*/
  (tIsrFunc)0,                                                 /* DLC 0xFFC2
*/
  (tIsrFunc)0,                                                 /* SCME 0xFFC4
*/
  (tIsrFunc)0,                                                 /* CRG Lock 0xFFC6
*/
  (tIsrFunc)0,                                                 /* Pulse AccB Overflow
0xFFC8    */
  (tIsrFunc)0,                                                 /* Mod Down Cnt
Underflow 0xFFCA */
  (tIsrFunc)0,                                                 /* PortH Interrupt
0xFFCC        */
  (tIsrFunc)0,                                                 /* PortJ Interrupt
0xFFCE        */
  (tIsrFunc)0,                                                 /* ATD1 0xFFD0
*/
  (tIsrFunc)0,                                                 /* ATD0 0xFFD2
*/
  (tIsrFunc)0,                                                 /* SCI1 0xFFD4
*/
  (tIsrFunc)0,                                    /* SCI0 0xFFD6                   */
  (tIsrFunc)0,                                                 /* SPI0 0xFFD8
*/
  (tIsrFunc)0,                                                 /* Pulse AccA Input Edge
0xFFDA  */
  (tIsrFunc)0,                                                 /* Pulse AccA Overflow
0xFFDC    */
  (tIsrFunc)0,                                                 /* Timer Overflow 0xFFDE
*/
  (tIsrFunc)0,                                                 /* Timer 7 0xFFE0
*/
  (tIsrFunc)0,                                                 /* Timer 6 0xFFE2
*/
  (tIsrFunc)0,                                                 /* Timer 5 0xFFE4
*/
  (tIsrFunc)0,                                                 /* Timer 4 0xFFE6
*/
  (tIsrFunc)0,                                                 /* Timer 3 0xFFE8
*/
  (tIsrFunc)0,                                                 /* Timer 2 0xFFEA
*/
  (tIsrFunc)0,                                                 /* Timer 1 0xFFEC
*/
  (tIsrFunc)0,                                                 /* Timer 0 0xFFEE
*/
```

```
  (tIsrFunc)RealTimeInterrupt,                                /* RTI 0xFFF0
*/
  (tIsrFunc)IRQInterrupt,                                     /* IRQ 0xFFF2
*/
  (tIsrFunc)0,                                                /* XIRQ 0xFFF4
*/
  (tIsrFunc)0,                                                /* SWI 0xFFF6
*/
  (tIsrFunc)0,                                                /* Unimpl Instr Trap
0xFFF8      */
  (tIsrFunc)0,                                                /* COP Failure
Reset(N/A) 0xFFFA */
  (tIsrFunc)0,                                                /* COP Clk Mon Fail(N/A)
0xFFFC   */
  (tIsrFunc)0                                                 /* Reset(N/A) 0xFFFE
*/
};


/************************************* end of vectors.c
*********************************/
```

16.4 Pixel

16.4.1 pixel.cpp

```
/*This is the main code that will drive the camera, map, and the sending the
path and directioins
  to the robot microcontroller.  It first uses a system call to execute
Map.exe which will generate
  a map and will wait for input from the user to generate a path to that room
number that was taken
  as input.  The path will then be converted into a vector of directions for
the micro-controller and
  a vector of strings for the camera code to be able to pan to catch the
correct door.  Then as the
  camera detects a door it will notify the micro-controller by calling an
executable called serial.exe.*/

//Program: Jeeves Mapping panning and moving
//Authors: Ramiro Diaz and Genaro Velasquez
//Major: Computer Engineering
//May 26, 2008

#include "Mapping.h"
#include "camera.h"
#include "functions.h"
#include <windows.h>
#include <vector>
#include <string>
#include<fstream>
```

```
using namespace std;

//This function is the camera panning
void camera(vector<Node*> campath)
{
	//is a door or not 1 yes 0 no
	int door_y_n = 0;

	//creates an instance of a Node object.
	Node curr;

	//will contain the door numbers
	vector<string> isdoortf;

	//This will fill the isdoortf vector with strings
	for(unsigned int i = 0; i< campath.size(); i++)
	{
		isdoortf.push_back(curr.getroom(campath[i]));
	}

	//will make the camera reset itself to look forward
//	door_y_n = look_door("Home");

	//This will go through the vector of doors and pan the camera
	//to its correct direction
	for(unsigned int ind =1; ind < isdoortf.size(); ind++)
	{
		door_y_n = look_door(isdoortf[ind]);

		if(door_y_n == 1)
		{
			door_y_n = 0;
			cout<<ind <<" : "<<isdoortf[ind]<<" :IT is a Door\n";
			//This system call will execute the serial.exe
			system("C:\\project\\serial.exe");
			door_y_n = 0;
		}
		else
		{
			cout<<"Not a door\n";

		}
	}
}

//This function is to decode the output of the map.exe from genaro
string location(int cur)
{
  string here = "";

  if(cur == 1){
    here = "Home";
  } else if(cur == 2){
    here = "136";
```

```
  } else if(cur == 3){
    here = "135";
  } else if(cur == 4){
    here = "inter1";
  } else if(cur == 5){
    here = "133";
  } else if(cur == 6){
    here = "132";
  } else if(cur == 7){
    here = "inter2";
  } else if(cur == 8){
    here = "129";
  } else if(cur == 9){
    here = "128";
  } else if(cur == 10){
    here = "127";
  } else if(cur == 11){
    here = "126";
  } else if(cur == 12){
    here = "inter3";
  } else if(cur == 13){
    here = "125";
  } else if(cur == 14){
    here = "Restrooms";
  } else if(cur == 15){
    here = "137";
  } else if(cur == 16){
    here = "inter4";
  } else {
    here = "";
  }

  return here;
}

/* Main program code.  Driver */
int main()
{
      //This is the instance of the Mapping object
      Mapping floor;
      //This is the vector for the node path for the micro-controller
      vector<int> Nodepath;
      //vector of nodes for the camera code
      vector<Node*> campath;
      //This is the key to determine what room was given by the user to look
for
      int mapout = 0;
//where i start new code from other file
      string valnum = "";
      int temp = 0;
      string sendserial="C:\\project\\serial1.exe ";
      string send = "";
      int pts = look_door("Home");
//--------------------------------------
```

```
      //While loop to keep asking for a room number
      while(1)
      {
            //This system call will execute the Map.exe
            //and save the ouput to a readme.txt
            system("C:\\project\\Map.exe > readme.txt");

            /*This part of the code will read from file*/
            ifstream myfile;
            myfile.open("readme.txt");
//code that was inputed when crazy
            myfile >> valnum;
            send = sendserial + valnum;
            system(send.c_str());
            temp = atoi(valnum.c_str());
            for(int i = 0; i < temp; i++)
            {
                  myfile >> valnum;
                  send = sendserial + valnum;
                  system(send.c_str());
            }
            myfile>>mapout;

//----------------------------------


            /*
            myfile >> mapout;*/
            myfile.close();
            //will contain the room number

          string place = location(mapout);
            //gets the path of directions.
            vector<int> path = floor.dij(floor.home, place, Nodepath,
campath);
            //calls the code for the camera to pan to its correct direction
            camera(campath);
  }

  return 0;
}
```

16.4.2 eyes.h

```
#ifndef __EYES_H
#define __EYES_H

//#include "stdafx.h"
#include <cv.h>
#include <cxcore.h>
#include <highgui.h>
#include<iostream>
//#include<stdio.h>
#include<string>
```

```
using namespace std;

int look_door()
{
	IplImage *frame = NULL;
	int key = 0;

	CvCapture* capture = cvCaptureFromCAM(0);
	if (capture == NULL)
	{
		printf("\nError on cvCaptureFromCAM");
		return -1;
	}
	cvNamedWindow("Image:",1);

	cvMoveWindow("Image:", 50, 50);
	while (true)
	{
		cvGrabFrame(capture);

		frame = cvRetrieveFrame(capture);

		cvShowImage("Image:", frame);
			//will get the size
		//---------------------------------------
		CvSize tmp;//this will be hight, width
		tmp= cvGetSize(frame);
		cout<<tmp.height <<", "<<tmp.width<<endl;

		CvScalar s;

		for(int i = 0; i < tmp.height; i++)
		{
			for(int j = 0; j < tmp.width; j++)
			{

				s = cvGet2D(frame,i,j);

				if((s.val[2]>= 155 && s.val[2]<= 170)&&(s.val[1]>=170 
&& s.val[1]<=185)&&(s.val[0]<55))
				{

					cvSaveImage("pixelfound.jpg",frame);
					cout<<"You found the pixel"<<endl;
					cout<<"("<<s.val[2]<<", "<<s.val[1]<<", 
"<<s.val[0]<<")"<<endl;
					cvDestroyWindow("Image:");
					cvReleaseCapture(&capture);
					return 1;
					//}
				}

			}
		}
```

```
            key = cvWaitKey(1);

            if (key==0x1b) // break if user hits ESC key
            {
                  break;
            }

      }
      //will close the window
      cvDestroyWindow("Image:");
      //cvReleaseImage(&frame);
      cvReleaseCapture(&capture);

      return 0;

}

#endif
```

16.4.3 camera.h

```
/*This part of the code will do some of the camera panning, tiling, and
zooming*/
//Some of the code is provided from opencv and from the Logitech company.

/*****************************************************************************
 * DirectShow Pan/Tilt/Zoom sample for Logitech QuickCam devices
 *
 * Copyright 2007 (c) Logitech. All Rights Reserved.
 *
 * This code and information is provided "as is" without warranty of
 * any kind, either expressed or implied, including but not limited to
 * the implied warranties of merchantability and/or fitness for a
 * particular purpose.
 *
 * Version: 1.1
 ****************************************************************************/

//Some parts have been changed and code has been added for our own use
//Ramiro Diaz & Genaro Velasquez
//You can get more info. on us at www.ee.ucr.edu/~radiaz

#ifndef __FNC_H
#define __FNC_H

#include <cv.h>
#include <cxcore.h>
#include <highgui.h>
#include<iostream>
#include <vector>
#include<string>
#include <dshow.h>
#include <Ks.h>                     // Required by KsMedia.h
#include <KsMedia.h>          // For KSPROPERTY_CAMERACONTROL_FLAGS_*
```

```
using namespace std;

int rreturn = 0;

/*
 * Pans the camera by a given angle.
 *
 * The angle is given in degrees, positive values are clockwise rotation (seen
from the top),
 * negative values are counter-clockwise rotation. If the "Mirror horizontal"
option is
 * enabled, the panning sense is reversed.
 */
HRESULT set_mechanical_pan_relative(IAMCameraControl *pCameraControl, long
value, int wvalue)
{
	HRESULT hr = 0;
	long flags = KSPROPERTY_CAMERACONTROL_FLAGS_RELATIVE |
KSPROPERTY_CAMERACONTROL_FLAGS_MANUAL;

	hr = pCameraControl->Set(CameraControl_Pan, value, flags);
	if(hr != S_OK)
		fprintf(stderr, "ERROR: Unable to set CameraControl_Pan property
value to %d. (Error 0x%08X)\n", value, hr);

	// Note that we need to wait until the movement is complete, otherwise
the next request will
	// fail with hr == 0x800700AA == HRESULT_FROM_WIN32(ERROR_BUSY).
	//Sleep(500);
	Sleep(wvalue);

	return hr;
}

/*
 * Tilts the camera by a given angle.
 *
 * The angle is given in degrees, positive values are downwards, negative
values are upwards.
 * If the "Mirror vertical" option is enabled, the tilting sense is reversed.
 */
HRESULT set_mechanical_tilt_relative(IAMCameraControl *pCameraControl, long
value)
{
	HRESULT hr = 0;
	long flags = KSPROPERTY_CAMERACONTROL_FLAGS_RELATIVE |
KSPROPERTY_CAMERACONTROL_FLAGS_MANUAL;

	hr = pCameraControl->Set(CameraControl_Tilt, value, flags);
	if(hr != S_OK)
		fprintf(stderr, "ERROR: Unable to set CameraControl_Tilt property
value to %d. (Error 0x%08X)\n", value, hr);
```

```
	// Note that we need to wait until the movement is complete, otherwise
the next request will
	// fail with hr == 0x800700AA == HRESULT_FROM_WIN32(ERROR_BUSY).
	Sleep(500);
	//Sleep(100);

	return hr;
}
/*
 * Resets the camera's pan/tilt position by moving into a corner and then back
to the center.
 */
void reset_machanical_pan_tilt(IAMCameraControl *pCameraControl)
{
	set_mechanical_pan_relative(pCameraControl, 180, 500);
	//Sleep(500);
	Sleep(500);
	set_mechanical_tilt_relative(pCameraControl, 180);
	//Sleep(500);
	Sleep(500);
	set_mechanical_pan_relative(pCameraControl, -64, 500);
	//Sleep(500);
	Sleep(500);
	set_mechanical_tilt_relative(pCameraControl, -24);
	//Sleep(500);
	Sleep(500);
}

/*
 * Test a camera's pan/tilt properties
 *
 * See also:
 *
 * IAMCameraControl Interface
 *     http://msdn2.microsoft.com/en-us/library/ms783833.aspx
 * PROPSETID_VIDCAP_CAMERACONTROL
 *     http://msdn2.microsoft.com/en-us/library/aa510754.aspx
 */
HRESULT test_pan_tilt(IBaseFilter *pBaseFilter, string path)
{
	HRESULT hr = 0;
	IAMCameraControl *pCameraControl = NULL;
	long value = 0, flags = 0;
    string current = "Home";


	// Get a pointer to the IAMCameraControl interface used to control the
camera
	hr = pBaseFilter->QueryInterface(IID_IAMCameraControl, (void
**)&pCameraControl);

	//This part of the code is to pan either righ or left depending
	//on the room number.
	if(path == "Home")
```

```
    {
        set_mechanical_pan_relative(pCameraControl,180, 500);
        reset_machanical_pan_tilt(pCameraControl);
        current = "Home";
        return 0;
    }
    if(path == "inter1")
    {
        set_mechanical_pan_relative(pCameraControl,-50,100);
        current = "inter";
        return 0;
    }
    if(path == "inter2")
    {
        set_mechanical_pan_relative(pCameraControl,50, 100);
        current = "inter";
        return 0;
    }
    if(path == "inter4")
    {
        current = "inter";
        return 0;
    }
    if(path == "inter3")
    {
        cout<<"im in path == inter3\n";
        rreturn = 0;
        current = "inter";
        return 0;
    }

    //uses opencv code and variables to do some image capture
  IplImage *frame = NULL;
    int key = 0;

    //gets the camera to open
    CvCapture* capture = cvCaptureFromCAM(0);
    if (capture == NULL)
    {
        printf("\nError on cvCaptureFromCAM");
        return -1;
    }

    //window name and where it will be located.
    cvNamedWindow("Image:",1);
    int number = 50;
    cvMoveWindow("Image:", 50, 50);

    while(true)
    {
        //more panning code.
        if(path=="136")
        {
            set_mechanical_pan_relative(pCameraControl, 50,30);
```

```
            current = "136";
      }
      if(path == "135")
      {
            current="135";
      }
      if(path == "133")
      {
            set_mechanical_pan_relative(pCameraControl,-50,30);
            current = "133";
      }
      if(path == "132")
      {
            current = "132";
      }
      if(path == "129")
      {
            set_mechanical_pan_relative(pCameraControl,50,30);
            current = "129";
      }
      if(path == "137")
      {
            set_mechanical_pan_relative(pCameraControl,50,30);
            current = "Shop";
      }
      if(path == "Restrooms")
      {
            set_mechanical_pan_relative(pCameraControl,-100,30);
            current = "Restrooms";
      }
      if(path == "125")
      {
            set_mechanical_pan_relative(pCameraControl,100,30);
            current = "125";
      }
      if(path == "126")
      {
            current = "126";
      }
      if(path == "127")
      {
            set_mechanical_pan_relative(pCameraControl,-100,30);
            current = "127";
      }
      if(path == "128")
      {
            set_mechanical_pan_relative(pCameraControl,100,30);
            current = "128";
      }

      while (true)
      {
            //gets a frame to process
            cvGrabFrame(capture);
```

```
                    frame = cvRetrieveFrame(capture);
                    //shows the frame to the user
                    cvShowImage("Image:", frame);

                    CvSize tmp;//this will be hight, width
                    tmp= cvGetSize(frame);//gets size
                    CvScalar s;
                    //for loop to process the image row by column
                    for(int i = 0; i < tmp.height; i++)
                    {
                          for(int j = 0; j < tmp.width; j++)
                          {

                                s = cvGet2D(frame,i,j);//gets the i and j pixel

                                //check to see if the pixel is the one we are
looking for
                                if((s.val[2]>= 146 && s.val[2]<=
170)&&(s.val[1]>=200 && s.val[1]<=240)&&(s.val[0]<70))
                                {
                                      //saves the image that found.
                                      cvSaveImage("pixelfound.jpg",frame);
                                      //destroys the window and releases the
camera

                                      cvDestroyWindow("Image:");
                                      cvReleaseCapture(&capture);
                                      //variable to return saying found camera
                                      rreturn = 1;
                                      cout<<"-----------------------\n";
                                      cout<<" Its returning with path = "
<<path<<endl;
                                      cout<<"-----------------------\n";
                                      return 1;
                                }

                          }
                    }
                    //waits to get key
                    key = cvWaitKey(1);

                    if (key==0x1b )// break if user hits ESC key
                    {
                          //will close the window
                          cvDestroyWindow("Image:");
                          //cvReleaseImage(&frame);
                          cvReleaseCapture(&capture);
                          break;
                    }

              }
        }
        return S_OK;
}
```

```
/*
 * Do something with the filter. In this sample we just test the pan/tilt
properties.
 */
void process_filter(IBaseFilter *pBaseFilter, string path)
{
	test_pan_tilt(pBaseFilter,path);
}


/*
 * Enumerate all video devices
 *
 * See also:
 *
 * Using the System Device Enumerator:
 *     http://msdn2.microsoft.com/en-us/library/ms787871.aspx
 */
int enum_devices(string path)
{
	HRESULT hr;

	//printf("Enumerating video input devices ...\n");

	// Create the System Device Enumerator.
	ICreateDevEnum *pSysDevEnum = NULL;
	hr = CoCreateInstance(CLSID_SystemDeviceEnum, NULL,
CLSCTX_INPROC_SERVER,
		IID_ICreateDevEnum, (void **)&pSysDevEnum);
	if(FAILED(hr))
	{
		fprintf(stderr, "ERROR: Unable to create system device
enumerator.\n");
		return hr;
	}

	// Obtain a class enumerator for the video input device category.
	IEnumMoniker *pEnumCat = NULL;
	hr = pSysDevEnum->CreateClassEnumerator(CLSID_VideoInputDeviceCategory,
&pEnumCat, 0);

	if(hr == S_OK)
	{
		// Enumerate the monikers.
		IMoniker *pMoniker = NULL;
		ULONG cFetched;
		while(pEnumCat->Next(1, &pMoniker, &cFetched) == S_OK)
		{
			IPropertyBag *pPropBag;
			hr = pMoniker->BindToStorage(0, 0, IID_IPropertyBag,
				(void **)&pPropBag);
			if(SUCCEEDED(hr))
			{
```

```
                        // To retrieve the filter's friendly name, do the
following:
                        VARIANT varName;
                        VariantInit(&varName);
                        hr = pPropBag->Read(L"FriendlyName", &varName, 0);
                        if (SUCCEEDED(hr))
                        {
                              // Display the name in your UI somehow.
                              wprintf(L"  Found device: %s\n",
varName.bstrVal);
                        }
                        VariantClear(&varName);

                        // To create an instance of the filter, do the
following:
                        IBaseFilter *pFilter;
                        hr = pMoniker->BindToObject(NULL, NULL,
IID_IBaseFilter,
                              (void**)&pFilter);
                        ////im here
                        process_filter(pFilter,path);

                        //Remember to release pFilter later.
                        pPropBag->Release();
                  }
                  pMoniker->Release();
            }
            pEnumCat->Release();
      }
      pSysDevEnum->Release();

      return 0;
}

int look_door(string path)
{
      rreturn = 0;
      int result;
      CoInitializeEx(NULL, COINIT_APARTMENTTHREADED);
      result = enum_devices(path);
      return rreturn;

}
#endif
```

Maglev: Magnetic Levitation Train

EE 175AB Final Report

Department of Electrical Engineering, UC Riverside

Project Team Member(s)	[redacted]
Date Submitted	6/8/2008
Section Professor	Sakhrat Khizroev

Summary

This report presents the final project on the experiment of magnetic levitation, experimented by Truyen Mai and Eugene Change

Fig. 1 – Maglev Project

Table of Contents

1 Executive Summary ([redacted])

The name of our project is an abbreviation of two words: magnetic levitation. The overall goal of our project is to design a public transportation system using magnetic levitation. There are two design objectives in our project. Our first design objective is to design the maglev system so that it will run smoothly around the track, which concerns stability. Our second design objective is to design the maglev system to achieve certain amount of performance within a range of specification.

We target our design mainly for developed and developing countries that need a new form of public transportation, or countries that suffer from excessive use natural oil. As more countries adapt to Maglev designs for their public transportation, it will become available to more countries around the world as the cost for the entire project reduces.

We implemented a miniature model as proof of our concept and design. We built a track using permanent magnet tiles. We first built a test train and propel the train using an electric motor and propeller. We then built our main design and propel the train and propel it using electromagnets. We also attempted an alternative design, but we were not able to implement it due to power constraint.

In the implementation of our test and main design, both trains run smoothly along the track with no disruption, and they both have power consumptions at an acceptable level. The second design objective is partially met because our main design is not as stable as we wanted it to be.

2 Introduction

2.1 Design Objectives and System Overview ([redacted])

Our project is to design a miniature model of maglev train system, which includes both train and track. Our miniature train model must be able to run on the track without the help of any external force. It should also run smoothly on the track without any disruption. Another important thing is that our train should move with minimal amount of power.

The intended application for our train is for any countries that wants a new source of public transportation, or countries that suffer excessive use of natural oil. Because our trains doesn't contact with the track, the friction will be eliminated. Less friction means that the train will be able to travel at a much faster speed with same or less amount of power needed for the conventional trains today.

The list of technical design objectives:

The trains will levitate from the ground, and make no contact with the track.

The trains will run smoothly along the track.

The trains will consume less than or equal to 12v power supply.

Member's responsibility:

Project Manager: Nathan

Concept and Design: Nathan, Eugene

Materials: Nathan, Eugene

Track Construction: Nathan, Eugene

Test train construction: Eugene

Main design construction: Nathan

Testing: Eugene, Nathan

2.2 Development Environment and Tools ([redacted])

Hardware and software tools:

Basic tools – Hammer, pliers, nails, measuring tape, scissors.

Waveform Generator – Used to produce saw tooth wave as an input for the oscillation to toggle current supply between the coils.

Testing equipment:

Multi-meter – Used to measure both voltage and current running through the coils.

Stopwatch – Used for timed test runs of our models.

Measuring tape – Used for both measuring distance of travel and precision measurement during model construction.

2.3 Definitions and Acronyms ([redacted])

Maglev- Short for magnetic levitation; often used to describe magnetic levitation train.

Permanent Magnet- Any material that produces magnetic field. For our project we used grade N42 permanent magnets with 13,000 Gauss.

Electromagnet- An electromagnet is a special type of magnet in which the magnetic field is produced by conducting an electric current through it.

IEEE- Institute of Electrical and Electronics Engineers

IEEE-UCR- Institute of Electrical and Electronics Engineers Student Chapter at the University of California, Riverside

UCR- University of California, Riverside

3 Design Considerations ([redacted])

3.1 Assumptions

There are some assumptions we made when we were building our project. Both of our train models will only work on the track that will generate 35lbs (128.1N) of magnetic levitation force together with our train. The performance of our train wouldn't be as good if it is put on any other tracks.

The next assumption would be that our trains are connected with a stable power source that can generate enough current to propel our trains. We have two trains. The Air bullet will travel at 1.8km/hr when it is provided with 3.48 V and 0.61A with a DC power source. Our main design, Low Rider, will work when it is connected with a DC power source of 12V, 3.02A

3.2 Realistic Constraints ([redacted])

Power is the main constraint on our train. As mentioned in section 3.1, our train models wouldn't achieve desired performance if it is connected with power sources that don't generate the required power. The air bullet requires a power source that can output 3.48 V and 0.61A to run. Low Rider requires an HP 33120A Function Generator to run with additional of a power supply that can supply at least 3Amp of current for the coils.

Weight constraint is also an important issue of our project. We have found that low rider will not be able to move if it has a load over 2.35kg. If the load of the train is over 5.4 kg, then the train won't levitate.

3.3 System Environment and External Interfaces ([redacted])

Our test train was connected to a HP E3630A triple output DC power supply. Our main design requires a HP 33120A 15MHz function/arbitrary waveform generator to provide a saw tooth wave current to work properly.

3.4 Industry Standards ([redacted])

Specification of the permanent magnet for our project:

- N42 grade magnets
- 13,000 gauss
- 17.5lbs of pulling force.

Specification of the HP 3630A DC power supply for our project:

- 0 to 6 V with a maximum current of 1 to 2.5A
- 0 to 20 V and 0 to -20 V with a maximum current of 0.5A
- 35 W triple output
- Constant voltage (CV) and Current limit (CL) modes (±20V)
- Constant voltage (CV) and Current fold back (CF) modes (+6V)
- Low noise and excellent regulation
- Digital voltage and current meters
- Output tracking

Specification of the HP 33120A function generator for our project:

- 15 MHz sine and square wave outputs
- 12-bit, 40 MSa/s, 16,000-point deep arbitrary waveforms
- Sine, triangle, square, ramp, noise and more
- Direct Digital Synthesis for excellent stability
- Option 001: High-Stability Time Base and PLL

3.5 Budget and Cost Analysis ([redacted])

After a few meetings to formulate a basic outline and structure of our project, we initially estimated our total cost of the entire project to be $300. As you can see from the table below, by the time we finished with our project, we spent pretty close to our initial estimation. We had the privileges of using many of the tools available from Dan Giles' office. Dan also provided many electrical parts (resistor, switches, etc...) to us. Thus, we were able to save much of our funds on that. It is fortunate that we were able to keep within our estimated range consider it was only up to the two of us to come up with the funds to spend on our project.

Item Descriptions	Price
Magnets	$130.11
Wires	$15.86
Sensors	$27.13
Batteries	$8.00
Bolts, Screws, Nails	$24.39
Metal Rails & Covers	$35.67
Lumber	$57.47
Accessories	$6.24
Total	$304.87

3.6 Safety ([redacted])

Safety is one of our main concerns when we were designing our project. We firmly believe that the development of Maglev trains will be focusing on energy efficient and safety rather than performance in the near future. This is why we focus our design on power consumption and stability. Stability is also one of our main design objectives.

3.7 Documentation ([redacted])

We search online for all the real life and experimental design people have came up with. We discussed about the possibility of implementing each design into our project. Most of these online documents were copied and printed out for later reference. We also read many books for conceptual ideas and understanding the basic principles. The chapters in the book that we think is helpful were printed out, or written into our notes for later reference.

3.8 Global, Economic, Environmental and Societal Impact

(Eugene)

Magnetic Levitation, or maglev for short, is widely regarded as the future of public transportation by many people. The principle of Maglev train is to install electro and permanent magnets on the train and the tracks. The magnets with same polarity will repel each other, thus making the train "levitates" from the track, which means that the maglev train will have no contact with the track. The train will travel without any friction and resistance besides the air resistance, which makes maglev trains with faster speed and better energy efficiency than the conventional wheeled trains. If maglev becomes the popular way of building public transportation, then it will be a great leap forward in terms of economics, the infrastructure of the public transportation system, and environment.

(Truyen Mai)

One of the main problems we have to deal with today transportation system is the overcrowding issue. When was the last time you visit an airport or a train station without it being overcrowded? Almost never. We are at the age where walking is no longer feasible. Everyone relies on transportation system to stay mobile. Let us look at Manhattan for example. Most people do not own their own cars and the majorities rely on subways and taxi for daily transportation. Even so, with a high population, our current transportation can not meet that level of demand, resulting in an on-going crowded place throughout the day, everyday. However, with the technology of Maglev train system, the train can attain speed that conventional wheeled-train can never dream of. In essence, a faster transportation means people can travel more frequently and ultimately reduce the queue time of people waiting for their next ride. To sum it up, a faster and easier form of transportation will reduce the problem of overcrowding globally.

(Eugene)

Economic issue has always been a major obstacle of maglev train projects around the world. Because maglev train works entirely different from conventional trains, building maglev trains means changing the entire infrastructure of public transportation system.

For example, the maglev train in Shanghai, China, which covers a distance of 30 kilometers from Pudong Airport to metropolitan area of Shanghai, costs around 40 million USD for each kilometer of the track. With the amount of passengers per day and the price of each ticket, the Shanghai maglev train won't be able to cover its original project cost even at the end of its lifespan. This is why maglev designs in the future will be focusing on reducing the amount of magnets needed on track. The design will also focus on making the system more power efficient in order to lower the operational cost. A successful case is the maglev of Gimini in Japan. Where the train may not achieve the speed as fast as most maglev trains, but the cost of each kilometer of track is lowered to 10 million USD. Although a little bit of performance is sacrificed, the train has become much more economical. If the economic problems are solved, then those countries that already have maglev trains will have an easier time maintaining and updating their public transportation infrastructure. For those countries that currently don't have maglev as their way of public transportation, it will be easier and less time consuming for them to construct and build their own infrastructures of maglev public transportation.

(Truyen Mai)

Up until now, the technology has been out there for maglev systems to put into practice, but there are technical issues that have prevented maglev from being widely implemented. The most important factor has always been cost. The system would require massive investment in long stretches of railways that the train will operate on. However, if we can look pass the initial investment for maglev projects, we can see that maglev system will save more money in the long run since the investment costs are offset by lower operating cost because maglev use far less energy and require virtually no natural resources to maintain. Economically speaking, we will be able to reduce the cost of transportation when using maglev train system by directly reducing the cost of maintenance and trim down the reliance on natural resources.

(Truyen Mai)

The potential of Maglev are very considerable and the technology is just around the corner, it is without a doubt within our reach. It also has the potential to become a major component of future mass transportation. The nature of maglev and its efficiency and anticipated speed can become a huge advantage for us to utilize to address many global, economical, societal and environmental problems and issues. If we continue to invest our money and effort into this technology, perhaps one day we will see maglev trains travel through cities at speeds faster than those attained by current airplanes while retain all the conveniences and capacity of current rail system.

3.9 Contemporary Engineering Issues

(Truyen Mai)

Since the beginning of the 21st century, transportation has become more vital than it ever had before. Up until now, the enormous demand in transportation has been met by relying on conventional automobiles and airplanes. However, the increased reliance on our current transportation system place heavy strains on it. The effects of this pressure include highway congestion, airport overcrowding, heavy usage of limited oil supplies, and pollution. It could be too expensive for us to totally invent a new mean of transportation, but if we can reinvent and revitalize the concept of trains and use its efficient "track" system, we might be able to find an alternative, yet better form of transportation. The idea of magnetic levitation train (coded Maglev) has been floating around for decades, but not until recent did people start to build a working prototype model of it. However, as engineers progress through building a full functional Maglev, they run into many design problems. Many of the contemporary issue relates to Maglev nowadays lies within the design process.

(Eugene)

New material will also be needed for maglev as it become the more popular way of transportation. One such material is the super conductors. Super conductors are materials that can conduct electricity with no resistance, thus creating zero energy loss. So far the phenomena of superconducting only happens when certain metallic alloy is super cooled to the absolute zero temperature, which is not a very convenient way. Researchers have been working on super conducting materials. Another new material needed for the future of maglev is a new form of permanent magnets, which will be able to generate enough pulling and repelling force, but at a much cheaper cost. Since maglev track is covered with permanent magnets, the cost of industrial grade permanent magnets has been a serious issue when constructing the tracks. If these breakthroughs in material can be achieved, then the maglev train will be able to perform a much faster speed and become more power efficient at the same time.

(Truyen Mai)

Another important issue that engineers must overcome to make maglev train work is to be able to build long stretches of track with as less resources as possible. However, building a track even with state of the art material is a difficult task by itself. In maglev train system, linear motors must fit within or straddle their track over the entire length of the train. Therefore, track design is extremely difficult unless it is point-to-point in a straight line. The train would lose stability during curves if the track is not carefully design. The body of the train must align will with the curve sections of the track; the curves must be gentle and avoid camber, while maintaining the switches on the track as they are very long and can easily be broken in current if not cared for properly.

(Eugene)

Magnetic forces are very powerful forces that people shouldn't ignore, on the other hand, it is also hard to control, making it unpredictable and hard to work with. This is why that the concept of maglev have existed for at least three decades, and only a very few countries in the world actually use it as their way of public transportation. In order to make maglev a safe and stable, it requires a stable, high voltage power supply. Stable power supply is not just essential to power the train, but also create stability when it is traveling on the track at extremely high speed.

(Truyen Mai)

One of the main advantage gains from using maglev over conventional train is that maglev train system does no produce as much noise as normal train does at lower speed. Little to no noise occurs when maglev trains travel at tentatively slow speed. However, it is known from a study that at higher speed, around 500 mph or more, the train would make tremendous amount of noise due to air displacement in the air which gets louder as the train go faster. High speed maglev trains are 5 dB noisier than traditional trains. Therefore, wither it is the aero dynamic or internal design, it is important that engineers can overcome this problem to maximize the advantages of using a maglev train over conventional wheeled trains.

(Eugene)

There is little doubt in my mind that maglev will become the future of public transportation. It is true that maglev is not a brand new concept, and the many engineering problems and unpredictability have prevented it from being widely used and adapted. Although engineers in the past decades have put considerable amount of effort in to solving these problems, they are still limited by the current level of technology, material, and health hazard issues. But as the demand for a high energy efficient public transportation is growing higher, mankind will be pushed to solve all these existing contemporary engineering issues of maglev, and maglev will become a fast, energy efficient, and convenient way to travel in the near future.

4 Experiment Design and Feasibility Study

4.1 Experiment Design

4.1.1 Levitation Test ([redacted])

Fig. 2 – Levitate Air Bullet

Objective: To test if our permanent magnet can produce a strong repulsive force to lift our trains. Also, how much weight the magnet can support

Setup: Procedure: Our initial platform weighs is at .5 lbs, we want to see if our magnet can support that platform. If it can lift the platform without trouble, then we will add additional weight in small amount until we deem that the levitation distance is too low to be stable (anything above 1 inch of levitation is acceptable).

Expected Results: We expect our magnet to be strong enough to be able to support the initial weight of the platform and will be able to support up to at least an additional of 100% of extra weight until it reaches the 1 inch levitation mark.

4.1.2 Electromagnet Strength ([redacted]

Fig. 3 – Electromagnet Test

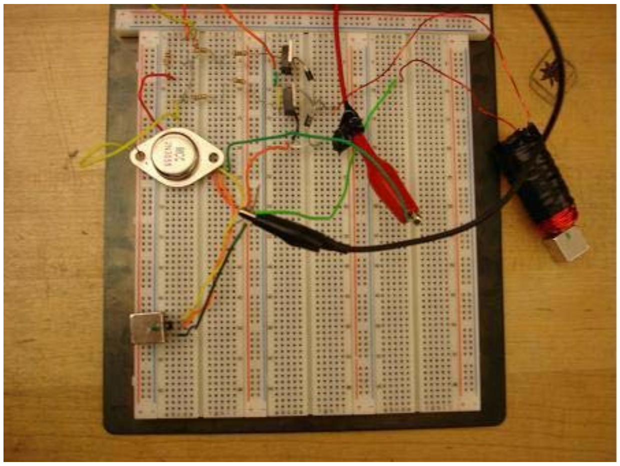
Objective: To find the appropriate current to apply to the coil to produce a magnetic field strong enough to negate the magnetic field produce by the permanent magnet.

Setup: In this component test, we wind an electromagnet with 16 rounds of wining around a 1'x ¼ Hex bolt with 21' gauge wires. We use a HP E3630A power supply to supply current to our coils. We use another permanent magnet to verify its strength. We measure so by looking at the repulsive force generate between the two magnets.

Procedure: We used on of our permanent magnet that we will use for the propulsion system to test the electromagnet. We will start at a low current and increase it until we find a current that can give the electromagnet a magnetic field that is strong enough to repel the field produced by the permanent magnet.

Expected Results: We expect a current of 2Amp should be enough to give the electromagnet a strong magnetic field

4.1.3 Circuit Test ([redacted])



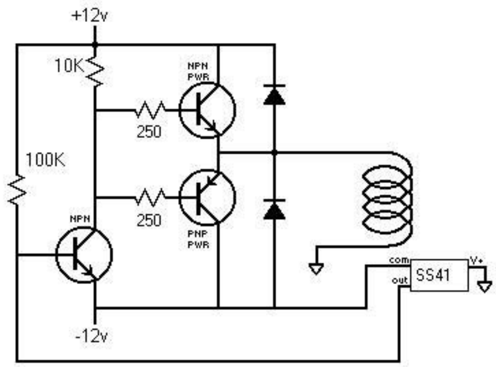


Fig. 4 – Circuit Setup and Schematic

Objective: To make sure the circuit can use the input from the Hall sensor to trigger the system to switch the polarity of the magnets when the sensor senses a magnetic field.

Setup: We built the above schematic on a circuit board. The two voltage input is being supplied by a bi-polar power supply. The circuit uses 2 NPN transistors and 1 PNP transistor. We used 2 rare earth magnet and a self-winded coil to test.

Procedure: Hook up the circuit from Fig. 4 and test to see if the circuit would switch the polarity of the electromagnet to the same pole that the sensor is sensing.

Expected Results: We expect the circuit to correctly switch the polarity of the electro-magnet accordingly to the same magnetic sensed by the SS41 sensor.

4.2 Experiment Results and Feasibility

4.2.1 Track levitation function test ([redacted])

After conducting this experience numerous times, we were quite please with what we discovered. Initially at .5 lbs, the magnets have no problem lifting the platform. We slowly increase the weight until it reaches 1 lb, which is the 100% mark that we expected. Even at 1 lb, the magnets were still able to lift the platform without any difficulties. So we decided to continue increase to test the limits of the magnet. We found out that around the 2.5 lbs mark, the platform starts to get really close to the track bed. It is true that the magnet can lift as much as close to 10 lbs of weight, but anything about 5 lbs make it extremely difficult to move the body of the train. We decided to stop the testing and conclude that as long as the train weights 2 lbs or less, the train should be able to move smoothly down the track. We were able to meet our expected results and reaches beyond the initial expectation. We concluded that the magnets are strong enough to use for our project.

4.2.2 Electromagnet Strength ([redacted])

This was a simple test. We simply placed a cube magnet (the stronger of the two types of magnets we bought) in front of our coil. We notice that as soon as the magnet comes close enough to the coil, the magnet would attract the iron core inside the coil. We started our test with 1A input into the coil, it proved to be too weak to fight off the pulling force between the iron core and the permanent magnet. We slowly made our way to 2 Amp, which was the initial expected value that the coils need. However, even at 2 Amp, the coils were still too weak. Thus, we continued to increase the current all the way to 2.5 Amp before the power supply reaches its peak current output. At this point, peaking with output current, and the magnet was still too weak, we decided to combine two power supplies together to increase the output current. We went from 2.5 Amp to 3.0 Amp and we finally reach a point where the coil produced a magnetic force that was strong enough to fight off the attraction between the iron core and the electromagnets. So we finally found out that 3.0 Amp is the minimum required Amp for the coil to be strong enough to work with the permanent magnet that we have. We continued to conduct more input current and both power supply seemed to cap out at 3.6Amp running at 12V (6V from each supply). At 3.6 Amp, the coil becomes a strong magnet but it runs extremely hot. We concluded that it was not safe and decided to use a current of 3.0-3.3amp as our safe input current. To conclude this experiment, we notice that our initial expected result failed because 2 Amp was not enough current to make a powerful magnet, at least not strong enough to work well with the permanent magnets we had for this project, at least 3.0 Amp is needed

4.2.3 Circuit Test ([redacted])

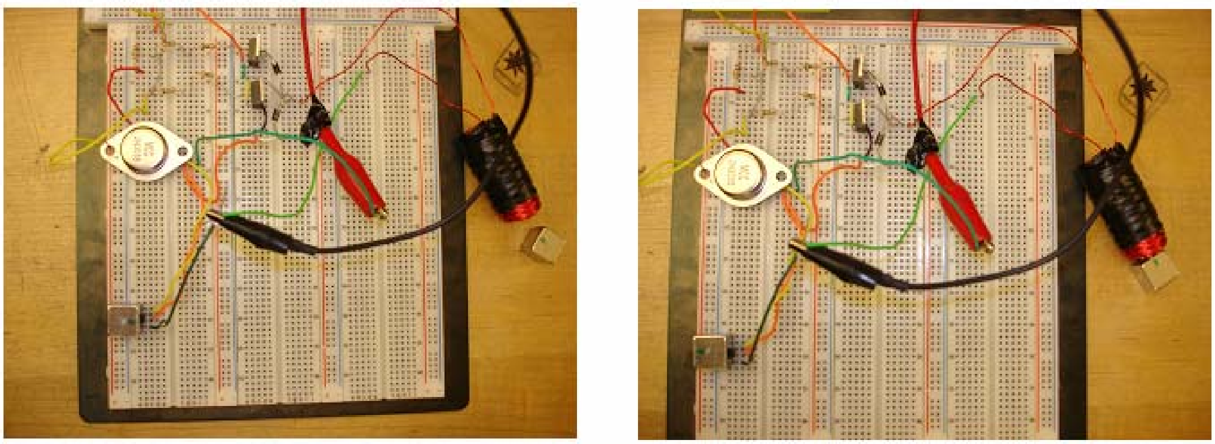


Fig. 5 – Circuit Test Results

After we hooked up the circuit according to the schematic in Fig. 4, it worked well. Note that the magnet used in this test is not an iron core based; it was made using a wood core. This was done because we simply wanted to test if the circuit was working properly, an iron core electromagnet would make this test a lot harder because it would require a massive amount of current (At least 3Amp) to test the circuit effectively. We use a wood core electromagnet to allow use to use smaller voltage and current input to test the circuit.

In Fig. 5 above include results for two separate tests. On the left, as you can see, the sensor is faced with the dotted side (N), the coils also switch the poles to N at the end and repel the other permanent magnet, this is an indication that the circuit work. To verify it one more time, we switch the magnet near the sensor with the opposite pole. As the result of that, the picture on the right show the permanent gets attracted to the electromagnet because the electromagnet at this point has switched to a different pole, which is the pole that the sensor had sensed. The circuit proved to serve its purpose and the result show that the circuit works perfectly.

However, from this experiment, we noticed that the electromagnet even though its pole was perfectly reversed, it lacked strength. At first we thought it was because of the wood core electromagnet, but we later found out that that was not the case. We didn't have to use an iron core to test the circuit because when we tried to use a higher voltage and current input, the max current that can be pulled from the power supply was .5 Amps, which is short from the required 3Am (look at 4.2.2). We came to Dan Giles for advice and told us that, because this design uses this circuit which require both positive and negative input, a bi-polar power supply would be needed. He informed us that UCR campus does not have any bi-polar power supply that is big enough to output 3Amp or more.

After knowing so, we concluded and decided to discard this circuit along with its design because we simply didn't have the hardware to implement our design. (This design is also listed in the Alternative Design Section of this report).

5 High Level Design

5.1 Conceptual View ([redacted])

After we found out that our first and second design cannot be implemented, we quickly did what we can and modified our second design (Sect. 7.2) to work with the materials we had at the time. Our final model basically contains two separate systems that work together to move the train. The block diagram in Fig. 6 indicates that there are the propulsion system and the levitation system. In our final design, the levitation system consists of permanent magnets that are placed on the side of the track and on the bottom of the train body. When they come close enough in contact, the natural magnetic forces will repel each other, and the repulsive force only gets stronger as the two set of magnets come closer together, thus levitate the train vertically.

However, to move the train horizontally, we needed a propulsion system to work with our levitation system. The propulsion system are composed of a DC current generator, a saw tooth wave generator, two self-winded coils, and another set of permanent magnets. The DC power supply would be connected to the electromagnet through a switch. The saw tooth wave generator would be connected to the NPN transistor switch, which acts as a gate to open/close the current from the DC power supply to the coil. The NPN gate is controlled and toggled by the signal output from the wave generator. With the saw wave signal controlling the NPN switch, the electromagnets on the train only receives current at specific intervals. With the switch constantly switching the electromagnets on and off rapidly, and at the same time reacting to the permanent on the track, the system make the train body to “jump” constantly which creates a wobble effect that help the train move horizontally.

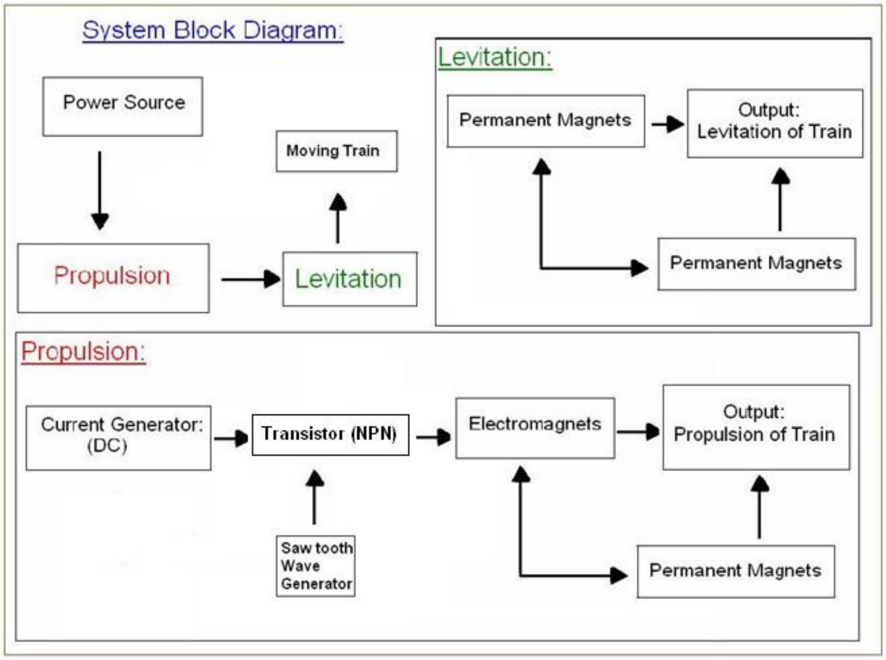


Fig. 6 High Level Block Diagram

5.2 Hardware ([redacted])

Fig. 7 - Track

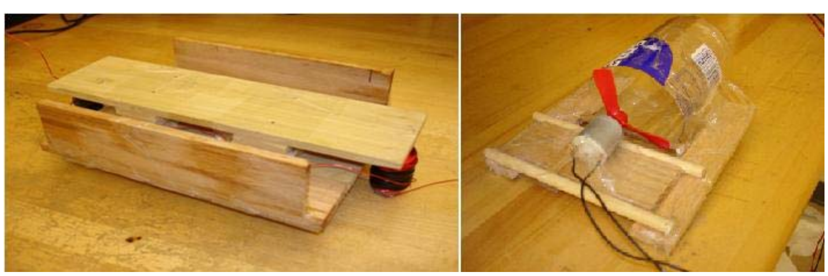


Fig. 8 – Train Models

Our project consisted entirely from the use of hardware. Our hardware includes two different modules: the two models of the train body (Fig. 8), and the track (Fig. 7).

First of all the track was built on top of a large platform of wood, we use this as our working platform and everything was build on top of it. The track itself contains two side walls that keep the train from falling off-track. They are both made out of wood as noted in Fig. 7. In between the walls lie three separate magnet tracks. The two tracks on the side contain strong rare earth magnet (Fig. 11), which are responsible for the levitation of the train body. The middle track, which is also constructed with rare earth magnet, is used strictly for our propulsion system. Each lanes, or rail, are covered by an aluminum hollowed stick. Inside the rails, the magnets are being attached to a metal stick. The metal stick serve as a binding site that help keep the magnet intact. Without the metal stick, the magnets can easily be moved when interacting with other magnets/electromagnets. The aluminum cover is placed on top to further contain the magnet in one place. Also, it makes the track look more professional by hiding the magnet well underneath the aluminum covers. Aluminum covers were used because they don't interfere or have any reaction with the magnetic fields.

The other major hardware component is the two different train models. Fig. 8 contains images of both models. The model with the red propeller is test model. It is made from thin wood. It has a small frame because the only thing that needs to be supported is the motor on top of it. The other model is a bit more complicated, it is also our main design for this project. It has a more complex frame because it needs to support two different electromagnets.

6 Low Level Design

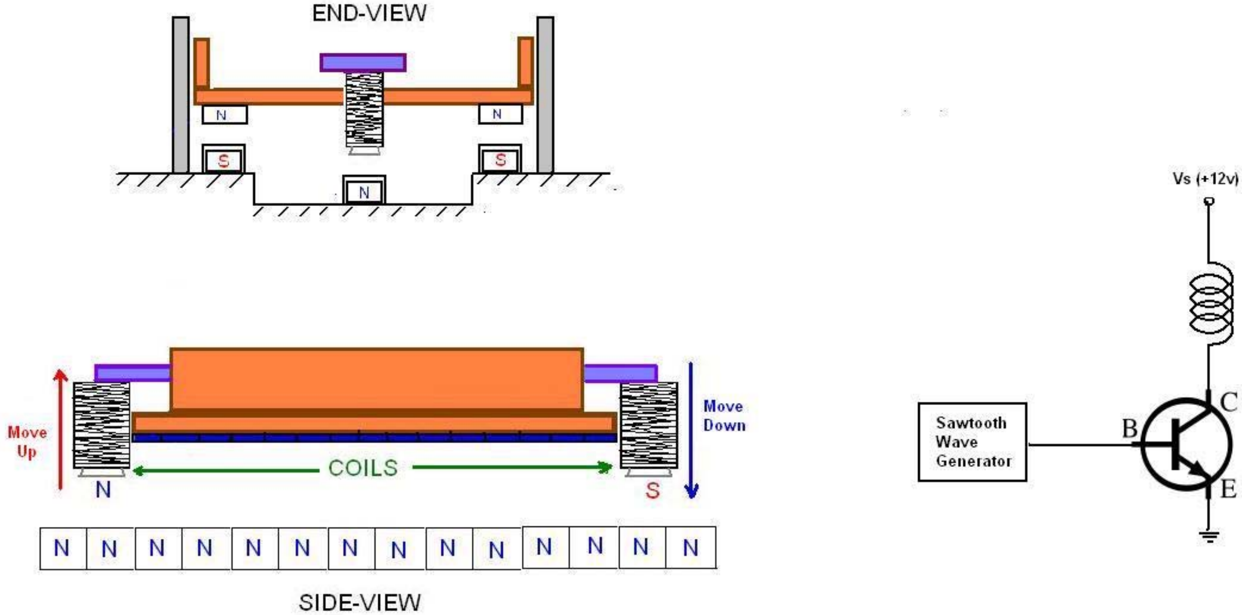


Fig. 9 – Detailed Design Diagram

6.1 Module 1: Coils (electromagnet) ([redacted])

We build and use our own coil throughout this project on several separate designs. The diagram above shows where these electromagnets are being used.

Fig. 10 – Coils

6.1.1 Module 1 Processing Details

Electromagnets are built with winding electrical wire around a center core. This core can be made from pretty much anything. A simple electromagnet can be made winding wires around a wooden stick, a plastic pen, or just have it hollow. However, electromagnets are most effective when the windings are being made around an iron core. A magnetic field can be generated by applying a current through the coil. The strength of the magnetic field strictly depends on the current applied, the number of winding, and the material of the core.

6.2 Module 2: Permanent Magnets ([redacted])

We mainly use permanent magnet to help with levitation. We placed them on both the trains and the track.

Fig. 11 – Permanent Magnets

6.2.1 Module 2 Processing Details

In Fig. 11 above, the thin rectangle magnets are used for levitation; they are placed inside the two lanes on the side. The cube magnets are stronger, and thus used for propulsion along with the electromagnets. These magnets have a magnetic flow through the surface.

6.3 Module 3: NPN Transistor (switch) ([redacted])

In our design, we use the NPN transistor as a gate to toggle and control the current going into our electromagnets on the train.

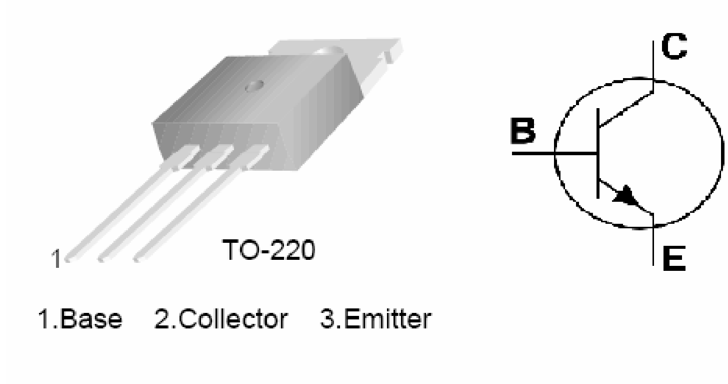


Fig. 12 – NPN Transistor and Symbol

6.3.1 Module 3 Processing Details

The NPN switch has three major components; Base, Collector, and Emitter. The base is what controlling the switch. The base turns the switch on when it receive a voltage signal of at least .7v or above. When the switch is turned on, current would be allowed to run from the Collector side to the Emitter end.

6.4 Module 4: Wave Generator ([redacted])

We use the wave generator to generate a saw tooth wave signal to toggle our transistor switch on and off.

Fig. 13 – HP Wave Generator

6.4.1 Module 4 Processing Details

The wave generator we used was the one found in the lab. It can produce many different wave forms which include: normal sine wave, square wave, burst, and saw tooth. User can also freely adjust from a wide range of frequency of oscillation along with different amplitude and offsets.

6.5 Module 5: DC Power Supply ([redacted])

We use these DC power supply to provide current for our electromagnets.

Fig. 14 – DC Power Supply

6.5.1 Module 5 Processing Details

This DC power supply is the one we use in our engineering labs. It is both a single and bi-polar power supply. The single positive mode supply only up to 6V of voltage but it can supply up to 2.5A of current in one channel. In bi-polar mode, it can supply up to +/- 20V of voltage but only up to .5A in each channel. We normally would combine two of these together running in single positive mode to bring 3+ Amp into our coils.

7 Alternative Designs

7.1 Alternative Design #1 ([redacted])

7.1.1 High Level Design

This design was the original design that we came up with in the very beginning. In this design, we decided to use massive electromagnets; we planned to use them for both levitation and propulsion. We wanted to use more electromagnets because permanent magnet tends to lose its effectiveness after a long period of time; therefore we wanted to design a system that relies on electromagnet to reduce maintenance costs in the long run. There are two separate systems in this design; Levitation and Propulsion system.

First of all, the Levitation system uses a DC power supply to supply current in to the electromagnets place on the track. When these magnets are turned on, as they react with the permanent underneath the train, the strong repulsive force will keep the train floated vertically.

In the propulsion system, instead of using a DC power supply, the design utilizes AC current for the electromagnets. In this design, many electromagnets will be placed on both sides of the tracks. An alternate current will constantly be flowing and switching the polarity of the magnets base on a specific frequency. As the poles are constantly switching among the electromagnets, the train will be pulled and pushed at the same time along the horizon axis. A better description of the motion can be seen in Fig. 16 below.

However, we later discarded this design due to it being too difficult for us to implement. As we started our construction of this design, we found it to be difficult and it cannot be achieved by the tools we had available to use. Because this design makes use of massive electromagnets through the tracks, every single individual electromagnet has to be consistently the same. That is hard to achieve because we had to make our own coil by hand. Without robotic precision, it proved impossible for us to consistently produce hundreds of electromagnets by hand.

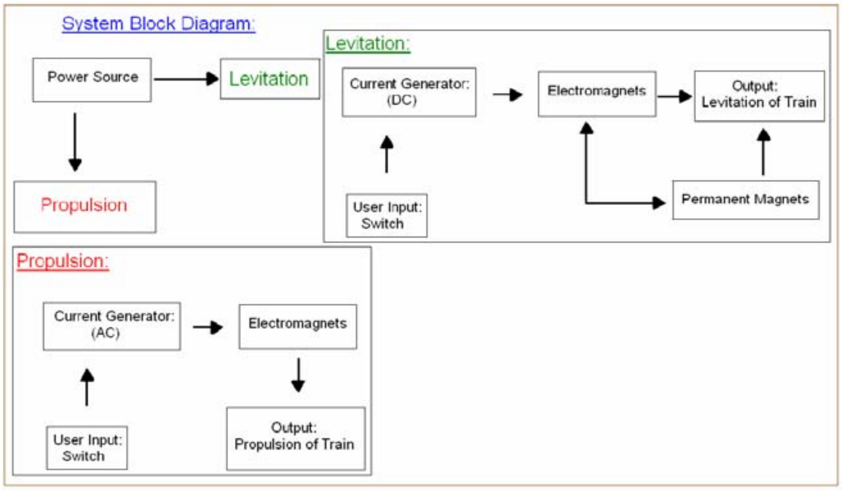


Fig. 15 – Alternative Design 1: High Level Block Diagram

7.1.2 Low Level Design

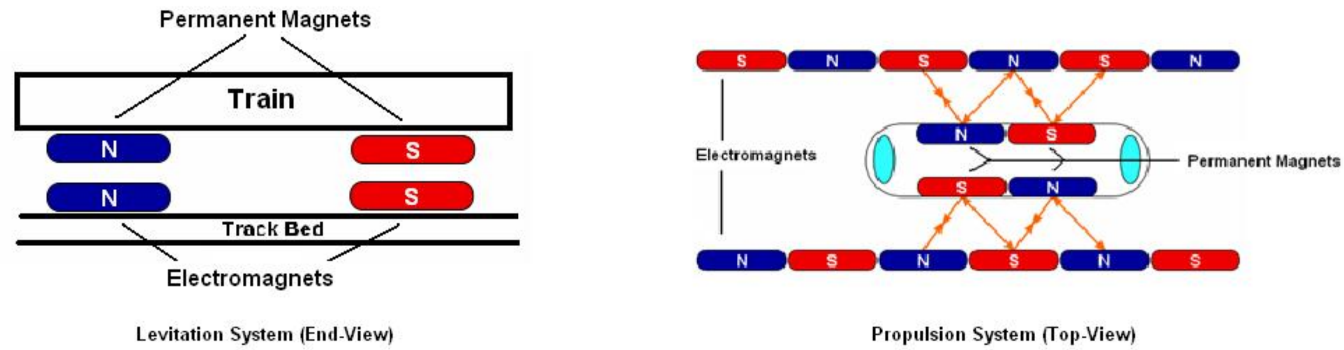


Fig. 16 – Alternative Design 1: Design's Details

7.1.2.1 Module 1: Electromagnet

See section 6.1

7.1.2.2 Module 2: Permanent magnet

See section 6.2

7.1.2.3 Module 3: AC Current Generator

We need to use an AC current generator to constantly switch the poles of the electro-magnets on the track to move the train horizontally.

7.1.2.3.1 Processing Details for Module 3

The AC current generator output both positive and negative current in a form of a wave. This is perfect to use on our electromagnet because we constantly need to change the polarity of our coils.

7.2 Alternative Design #2 ([redacted])

7.2.1 High Level Design

After concluding that our first design can implanted, we quickly researched and came up with a new plan. This plan was also discarded later after we found out about its requirements. However, it is a very intrigue design and it is worth mentioning.

Similar to our first design, this design also consist of two different systems (levitation and propulsion), which work together to move the train. The Levitation system becomes less complicated as we switched from using electromagnet to permanent magnets. We found permanent magnet to be more stable since we don't have to make them ourselves. Despite the similarity with using two separate systems, the propulsion in this design is completely different from our original design. This design is based on the technology often implemented in dc linear motors. The body of the train will be levitated using permanent attached to the bottom of the train. In the middle of the train, a few coils will be attached in a row (look at Fig. 18 for better visualization). Each of these coils is accompanied be a separate set of circuit (the schematic of the circuit can be seen in Fig. 18). On the track, the middle lane will be equipped with a long roll of permanent magnet, the magnet will be arranged with their poles alternating, and we called it the NSNS track.

With the arrangement of the permanent magnets on the track, the way the train move forward is that the sensor standing in front of the coil facing the track is constantly detecting the magnetic field around it. As soon as it detects a magnetic field, the system will switch the polarity of that coil to match that magnetic field so it will always repel. However, the important thing to note here is that since it is an NSNS track, the coil will always be attracted to the permanent magnet in front of it since the track arrange the magnet to alternate. Thus, the coil polarity will attract with the next magnet coming up and as soon as it gets there, the sensor would demand it to switch back and the cycle repeats again. Thus, if working perfectly, the train will move forward.

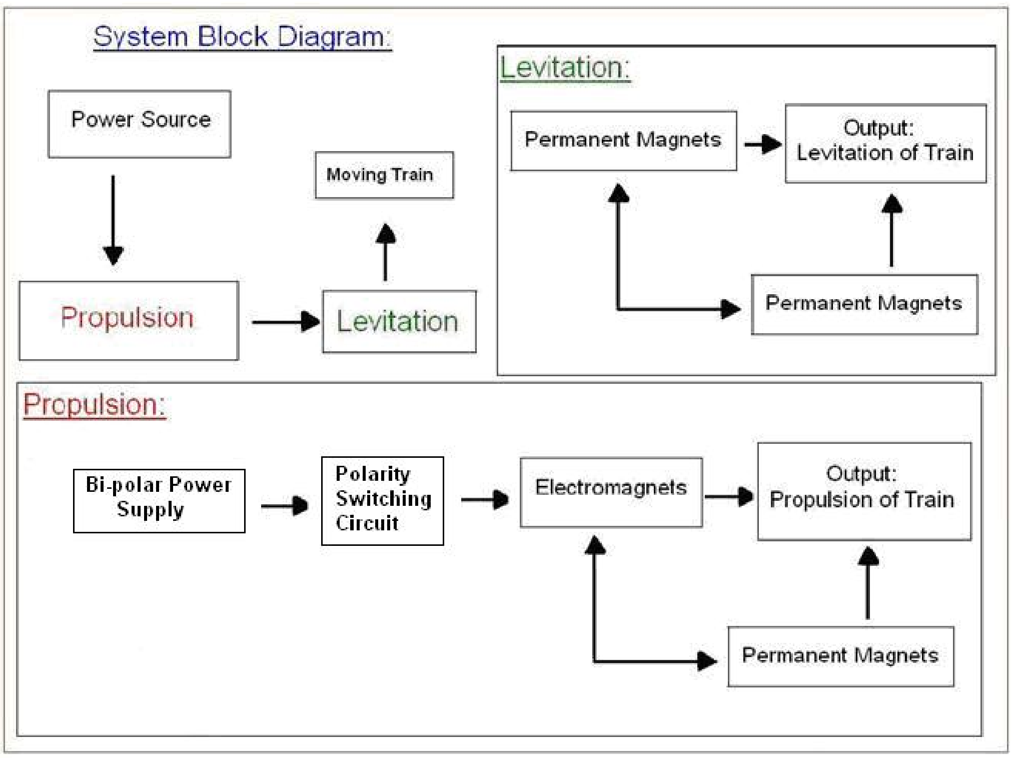


Fig. 17 – Alternative Design 2 High Level Block Diagram

7.2.2 Low Level Design

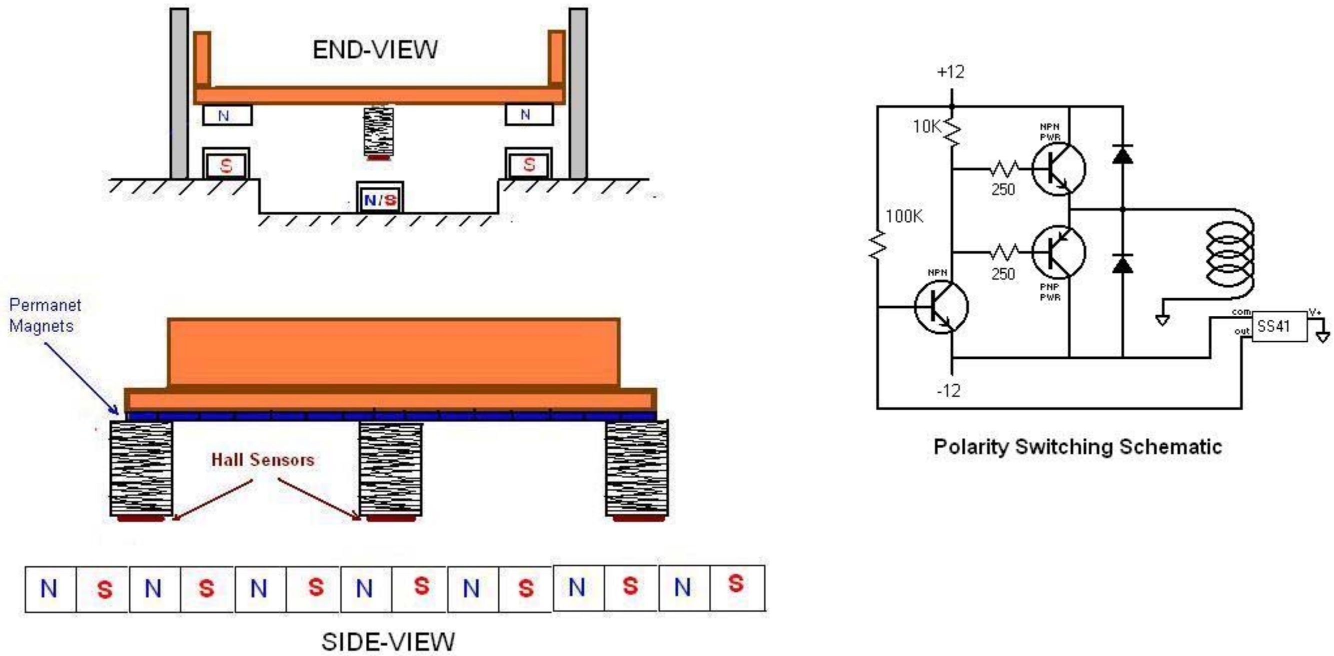


Fig. 18 – Alternative Design 2: Linear Motor Design’s Details

7.2.2.1 Module 1: Electromagnet

See sect. 6.1

7.2.2.2 Module 2: Permanent magnet

See sect. 6.2

7.2.2.3 Module 3: NPN Transistor

See sect. 6.3

7.2.2.4 Module 4: Bi-polar DC power supply

Bi-polar power supply can supply both positive and negative current into a system at the same time. We use this in our design to give us the flexibility to supply both positive and negative current into our coils.

7.2.2.5 Module 5: SS41 Hall Sensor

We use the SS41 in this design to detect a magnetic field that can trigger the system to switch the polarity of the electromagnet.

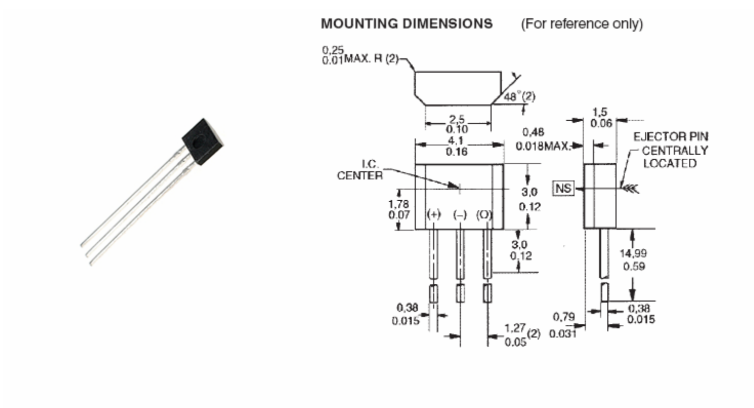


Fig. 19 – SS41 Hall Sensor Chip

7.2.2.5.1 Processing Details for Module 4

The SS41 Hall sensors have three different pins (voltage input, voltage output, and Ground). When the sensor detect a certain magnetic field, depending if it's north or south poles, the chip will send a small voltage signal out as output. This output is crucial as we will use it to trigger the circuit to allow it to switch the polarity of the electromagnets on the train so it can react accordingly to the permanent magnet placed on the track (only used in alternative design).

8 Administrative and Other Design Issues

8.1 Project Management

Truyen Mai: I am the project manager and I was responsible to schedule meetings between Eugene and I. The first 3 weeks, we met up twice a week to brainstorm and research for our project. We eventually meet up more often during the school week once we knew what we wanted to do. I divided up some of the work between me and Eugene, especially when it comes to research. However, as we combined our findings and pieced our design together, we often work together. For example, we both worked on the track because we both needed to know the track well to be able to build our own test models. Eugene was responsible for building a test model while I worked on our main design model. Toward the end of the project, we both did extensive testing and test runs together. Many of the things we did at that point were a group thing. It just didn't make sense for us to split up the work because there were only two of us and I felt that it was crucial that both of us know the details and understand everything that we did. Eugene was an excellent partner, we worked well together and it was a great experience for me.

Eugene: We worked on most of the project together at the beginning. We usually scheduled to meet up twice a week when we were working on the design and concept of our project. Later on Nathan did well dividing the work to acquire the materials we need to build our project. We both worked on the track, and the track was able to finish earlier than we expected. We later split the work up to save time. I worked on constructing and experimenting our test train, and Nathan was able to build our main design using the data and experience gathered from the experiment on test train. It was a pleasure to work with Nathan; he did a fine job as project manager.

9 Experiment Design and Test Plan

9.1 Design of Experiments

9.1.1 Experiment Case 1: Stability ([redacted])

1. Objective: Test the ability to move smoothly down the track for both models.

2. Setup: Both the Air Bullet and Low Rider will run on the same track.

3. Procedure: Each model will perform three trips, record and make any notes about their stability problem.

4. Expected Results: The Air Bullet should be able to move through the entire track without any trouble. The Low Rider, however, should not be as stable as the Air Bullet.

9.1.2 Experiment Case 2: Frequency ([redacted])

1. Objective: Find a suitable frequency of oscillation to move the Low Rider down the track.

2. Setup: With the Wave Generator, the power supplies, and the coil connecting through a switch, use the wave generator to find a suitable frequency.

3. Procedure: Have the Low Rider hooked up to the NPN switch and the power supply. Then connect the wave generator output to the NPN switch. Start off at 1 kHz frequency, start going up if the train wobbles too slowly or turn the frequency down if the train wobbles too fast.

Wave Generator Setup:

- Turn on the Wave Generator; the frequency should be at 1 kHz (default).
- Hit Amplitude button and set the amplitude to 5.00 VPP. This will result in a 1V peak to peak output. The NPN switch needs at least .7V to open its gate.

- Hit the offset and set it to +500 mV DC. The offset is use to filter out the negative signal, as we only need the positive signal output for the gate to open.
- Connect the signal to the oscilloscope to check to make sure the offset is correct and the peak to peak voltage is 1v. Fig. 20 shows a correct reading on the oscilloscope.

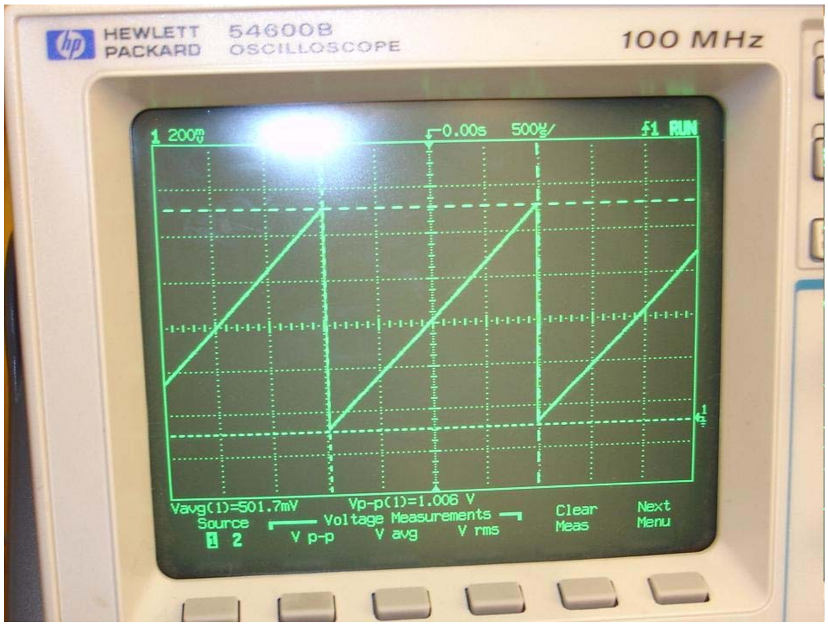


Fig. 20 – Correct Output reading of Wave Generator through an Oscilloscope

4. Expected Results: 1 kHz might be too fast, we expected 750Hz would be a good frequency to operate the train at.

9.1.3 Experiment Case 3: Speed & Time ([redacted])

1. Objective: Test the speed of both train models.

2. Setup: Using a stop watch to record the time it takes both model to finish running down the track (1 meter track).

3. Procedure: Each model will take three trips down the track, record their time and use their average time to calculate the speed.

4. Expected Results: The Air Bullet will take two seconds to complete the track while the Low Rider will take 7 seconds.

9.2 Quality Control ([redacted])

Each of the test trial we ran during our experiment was recorded. We paid additional attention to fail cases and usually try to correct the test or fix the problem that was obvious to improve the next test trial. Failures and problems during the test cases are to be discussed between both group members to formulate and come up with a plan to fix the problem.

9.3 Identification of critical components ([redacted])

Permanent Magnet – this component is very important to us. Since we use massive permanent to levitate our train, if one falls off or broke, it could result in a lot of instability of the moving train.

10 Experimental Results and Test Report

10.1 Test Result for Case 1: Stability ([redacted])

The result for this test indicates that the Air Bullet is very stable and it moves down the track smoothly for all three trials. Stability was not a problem. It met the expected result.

The Low Rider, on the other hand, took quite a bit of effort to move down the track. In the first trial, we ran the Low Rider with a frequency of 1 kHz. The fast frequency causes the train to wobble way too fast and the train only stayed in one place. The second trial, we dropped the frequency in half and ran it at 500 Hz. This creates a huge wobble affect since the train only "jumps" at slow interval. A big jump ended up crashing the electromagnet into the track. Fig. 21 below show a picture just right before the train was about to crash into the middle track lane.

Fig. 21 – Low Rider right before a crash

After the two trials, we concluded that the train was not able to run smoothly down the track with those random frequencies we threw at it. The result for this case did not meet up the expected result. We decided to do a frequency test for the next case to correct the movement of low rider.

10.2 Test Result for Case 2: Frequency ([redacted])

From the result in test case 1, we knew that the sweet spot is anywhere between 1 kHz and 500 Hz. Again, we started at 1 kHz again just to stay on the safe side. We did many trial runs here, each one for every time we want to test a new frequency.

As the test went on, the train still wobbled extremely fast from 1 kHz to 900 Hz, the train did not move at all. We slowly adjusted our frequency and once we got to 850, the train started to move slowly, very slowly, down the track. However, it was too slow so we decided to keep moving down in frequency. It got better on from here as we decrease the frequency. Finally, we arrived at 840 Hz and that frequency allowed the train to successfully run through the entire track.

We decided to continue lowering the frequency to 750 Hz, which was our expected result. However, as soon as we past the 810 Hz, the train start to wobbles slowly and it crashed into the track. The Low Rider had the best test run at 830 Hz

Seeing this, we realized that 830 Hz is the sweet spot for the Low Rider to run on. This was quite a bit far away from our predicted frequency of 750 Hz.

10.3 Test Result for Case 3: Speed & Time ([redacted])

Air Bullet:

Trial #	Time (sec)
1	1.9
2	2
3	1.9
Average	1.93

Low Rider:

Trial #	Time (sec)
1	14
2	12
3	15
Average	13.6

The two table above show all the trial test runs for both the Air Bullet and the Lower rider respectively. It is clearly shown here that the average speed of the Air Bullet is much faster comparing to the Low Rider. This is when both model are run at there optimal setup. The Air Bullet is moving with the motor running at max speed (8000 rpm) and the Low Rider running at its most stable frequency (830 Hz).

The Air Bullet finished at 1.93 sec for average speed, which is almost identical with the expected result. The Low Rider, however, did not quiet achieve what we expected it to do. It was almost two times slower than what we expected it to run at. This is possibly due to the weight of the train. We have already optimized the train body be as light as possible so we couldn't reduce anymore weight of the Low Rider to test it at a different mass.

11 Conclusion and Future Work

11.1 Conclusion ([redacted])

Our finished project meets most of the project goal. We were able to build a miniature model of transportation system using magnetic levitation. Both of our test models were able to run smoothly on the track without any disruption. Our main design was able to run along the track, but we believe that its stability was not good enough to be a safe design for carrying any passengers onboard. Our main design (Low Rider) did not run as smoothly as we wanted it to run. However, we were quite please that our design worked and it was implemented successfully.

We conclude that magnetic force is a very powerful force, and is a possible way of transportation in the future. On the other hand, maglev train is an attractive theory, but it was not easy to implement it. Although the idea of maglev train has existed for many decades, there are still too many problems people were not able to solve, and very few countries actually adapt it for their public transportation. The success in our test train; however, lead us to believe that it is possible to install a large wind turbine on a maglev train, and the train will be able to generate power as it runs, and reuse that power to keep it running, making the train a zero-power consumption design.

11.2 Future Work ([redacted])

If we had more time for our project, we would certainly expand and improve our project under these four concepts:

- Applying magnets on the side walls of the track to reduce the balance problem of our train, and make sure that there is no friction between train and side walls of the track.
- To modify our test train into a train that can use the wind to generate power as it runs, and reuse that power for the propulsion. This will make it a possible design with zero power consumption.
- To expand our track, include corners, inclines, etc. And make sure that our train works on these terrains.

- Acquire a better bipolar power supply (one that can supply 3A) to make our alternative design work. We truly believed that our second design was our best design but weren't able to implement it correctly because we lack the tool and hardware. We believe that with the right power supply, we can definitely make the design work the second time around.

11.3 Acknowledgement ([redacted])

We want to express our gratitude to the following people because we would never be able to finish our project without them: We want to thank our section professor, Prof. Sakhrat Khizroev, for his inspiration and professional assistance throughout this entire project. We hope that he will continue to inspire and help students like he did for us. We would like to thank Dan Giles for his assistance, advices, and providing the parts we didn't have. We hope that he will enjoy his retirement. We thank Imran Yousuf and his group for donating the propeller for our test train. We thank IEEE-UCR for letting us to use their equipment and facilities. Finally, we would also like to extend gratitude to M.S. Tso-Wei Lee of Texas A&M and Prof. Tsun-Kuo Chang of NTU for their brainstorming, encouragement, and professional advices.

12 References ([redacted])

[1] Beaty, William J. "Magnetic Levitation Cradle," [Online] Available: http://amasci.com/maglev/maglev.html

[2] Iskander, Magdy F., Electromagnetic *Fields and Waves*. Illinois, 2000, Waveland Press, Inc., pp.181-212

[3] Sedra, Adel S., Smith, Kenneth C., Microelectronic *Circuits, 5th Ed.* New York, Oxford, 2004.

13 Appendices ([redacted])

Appendix A: Parts List

Part Name	Description
Lumber Board	Used to make the base of our track
Aluminum Cover	Used to make the cover our track
Metal strips	used as foundation to place all the permanent magnet tiles for our track
Lumber	Used to built the side walls and train bodies
Magnet Tile	Dimension: 1''x0.5''x0.25'', Gauss: 13,200, Pulling Force: 17.5lbs
Magnet Cube	Dimension: 1'' Cube, Gauss: 13,200, Pulling Force: 25.09lbs
Electric Motor	Electric Motor: 8000RPM at 3.0 VDC
Funnel	Air Funnel used to produce more thrust for the test model, made from plastic bottle
Cardboard	Used for some part of the train bodies.
Various Resistors, diodes, and transistors	Some from EE1A/B and EE100A/B kit, rest were donated by Dan Giles
Propeller	Donated by Imran Yousuf
Metal bolts	5/16'' bolt, 3/2'' long bolts used to construct our electromagnets
Magnet Wire	21 gauge wire we used for our electromagnets

Appendix B Equipment List: ([redacted])

- HP 54600 B Oscilloscope
- HP 34401 A Digital Multi Meter
- HP 33120 A Function Generator
- HP E3630A Triple Output DC Power Supply