Final Design Report

Team 09: In The Spotlight

Phil Fynan, Jake Roorda, Lucas tenBrink, Khoi M. Vu

Engr339/340 Senior Design Project

Calvin College
15 May 2014
Executive Summary

As part of Calvin College’s Senior Design course, the members of team 9 planned to create a power-efficient, robotic spotlight that automatically tracks a stage actor in theaters and other similar venues. Current spotlights require dedicated human operators and waste a significant amount of power as heat. In order to address these problems, the robotic spotlight designed by the team autonomously tracks a target after being selected in the graphical user interface (GUI) and utilize an LED light source to reduce power consumption. A parabolic reflector and light filter attachment optical design reduces costs by eliminating the need for lenses, and a mirror was to direct the light instead of directing the spotlight itself to reduce the weight the motors must move. The prototype design cost $775 and took over 850 working hours to prototype. The final prototype system was demonstrated at the Calvin College Senior Design Open House on May 10. At the open house, the light was able to track the beacon with sufficient accuracy to keep the user in the beam of light at all times. Though, near the end of the demonstration, sunlight from one of the windows proved to be a distraction for the tracking system. That said, later nighttime tests, including one linked from http://www.calvin.edu/academic/engineering/2013-14-team9/UpdatesPage.html, showed the light performing as desired.
Table of Contents

Executive Summary ........................................................................................................... i

Table of Contents ........................................................................................................... ii

Table of Figures ............................................................................................................. viii

Table of Tables ............................................................................................................. xi

1 Introduction .................................................................................................................. 1
   1.1 Context ...................................................................................................................... 1
   1.2 Proposal ................................................................................................................... 2
   1.3 Team Members ....................................................................................................... 2
      1.3.1 Phillip Fynan ..................................................................................................... 2
      1.3.2 Jake Roorda ..................................................................................................... 2
      1.3.3 Lucas tenBrink ............................................................................................... 3
      1.3.4 Khoi M. Vu ..................................................................................................... 3
   1.4 Design Norms ......................................................................................................... 3
   1.5 Report Structure ..................................................................................................... 3

2 Project Management .................................................................................................... 5
   2.1 Team Management ................................................................................................ 5
   2.2 Division of Labor ................................................................................................... 7
   2.3 Schedule ............................................................................................................... 8
   2.4 Operational Budget ............................................................................................. 10

3 Objectives and Requirements .................................................................................... 14
   3.1 User Interface Requirements ............................................................................ 14
      3.1.1 Centralized Control ......................................................................................... 14
      3.1.2 Manual Direction Functionality ..................................................................... 14
      3.1.3 Simplified Control ......................................................................................... 14
      3.1.4 System Response to User Input ..................................................................... 15
      3.1.5 Interaction Demand during Performances ................................................... 15
   3.2 Installation Requirements .................................................................................... 15
3.2.1 Spotlight Mounting
3.2.2 Conspicuousness
3.2.3 Flexibility

3.3 Communications Requirements
3.3.1 Facility Data Transfer Cable Compatibility

3.4 Safety Requirements
3.4.1 Electrical
3.4.2 Mounted Components
3.4.3 Obstructions
3.4.4 Materials

3.5 Power Requirements
3.5.1 Energy Use
3.5.2 Power Source

3.6 Performance Requirements
3.6.1 Tracking Accuracy
3.6.2 Lighting Conditions
3.6.3 Tracking Subject Loss
3.6.4 Environmental Conditions
3.6.5 Actuation
3.6.6 Beam Edge Quality
3.6.7 Beam Radius on Subject
3.6.8 Noise
3.6.9 Illuminance on the Stage
3.6.10 Light Output
3.6.11 Color Temperature

3.7 Deliverables
3.7.1 PPFS Report
3.7.2 Working Prototype
3.7.3 Final Report
3.7.4 Business Plan
3.7.5 Software
3.7.6 Hardware Specifications
3.7.7 Design Documentation
6.3.3 Heat Sink .................................................................................................................. 50
6.3.4 Controller .................................................................................................................. 52
6.3.5 Spotlight Mounting ..................................................................................................... 54
6.3.6 Parabolic Reflector ..................................................................................................... 56

6.4 Lighting Subsystem Software ..................................................................................... 58

6.5 Testing and Debug ....................................................................................................... 60
6.5.1 Subsystem Testing ..................................................................................................... 60
6.5.2 Debug ...................................................................................................................... 64

7 Sensing Subsystem ........................................................................................................ 66
7.1 Camera System ........................................................................................................... 66
7.1.1 Development Board .............................................................................................. 66
7.1.2 Types of Camera ..................................................................................................... 68
7.1.3 Design Decision ...................................................................................................... 70
7.1.4 Video Streaming Prototype .................................................................................... 72

7.2 Tracking Beacon ......................................................................................................... 74
7.2.1 Research .................................................................................................................. 75
7.2.2 IR Beacon in Spotlight tracking ............................................................................. 75
7.2.3 Initial IR Beacon Design ........................................................................................ 75
7.2.4 Final Beacon Design .............................................................................................. 77

8 Tracking Subsystem ...................................................................................................... 81
8.1 Requirements .............................................................................................................. 81
8.2 Decision Criteria ......................................................................................................... 81
8.3 Decision Alternatives ................................................................................................. 81
8.3.1 Algorithm ................................................................................................................ 81
8.3.2 Hardware ................................................................................................................ 82
8.3.3 Software .................................................................................................................. 83

8.4 Design decision .......................................................................................................... 84
8.4.1 Hardware ................................................................................................................ 84
8.4.2 Software .................................................................................................................. 84
8.4.3 Algorithm ................................................................................................................ 84

8.5 Implementation Details .............................................................................................. 85
9 User Interface Subsystem ............................................................................................................. 88
  9.1 Research .................................................................................................................................. 88
  9.2 Control Overview ....................................................................................................................... 88
  9.3 Hardware ................................................................................................................................... 88
    9.3.1 Hardware Requirements ........................................................................................................ 89
    9.3.2 Operating System .................................................................................................................. 89
  9.4 Software Design .......................................................................................................................... 89
    9.4.1 Programming Language ........................................................................................................ 89
    9.4.2 Software Overview ................................................................................................................. 89
    9.4.3 Communication ..................................................................................................................... 89
    9.4.4 User Interface Design ............................................................................................................ 93
    9.4.5 Design Outcomes .................................................................................................................. 97
10 Actuation Subsystem ...................................................................................................................... 98
  10.1 Research ................................................................................................................................... 98
  10.2 Preliminary Testing .................................................................................................................... 99
  10.3 Overall Design Alternatives ...................................................................................................... 100
    10.3.1 Decision .............................................................................................................................. 103
    10.3.2 Implementation .................................................................................................................... 104
  10.4 Actuation Hardware .................................................................................................................... 105
    10.4.1 Frame Design ....................................................................................................................... 105
    10.4.2 Power Transmission .............................................................................................................. 105
    10.4.3 Supports ............................................................................................................................. 108
    10.4.4 Motor Alternatives ................................................................................................................. 110
    10.4.5 Mirror ................................................................................................................................. 112
    10.4.6 Electronic Hardware ............................................................................................................. 113
  10.5 Actuation Software ..................................................................................................................... 124
  10.6 Communications ......................................................................................................................... 126
  10.7 Testing and Debug ....................................................................................................................... 127
11 System Integration ......................................................................................................................... 130
  11.1 System Level Test ..................................................................................................................... 130
  11.2 System Level Debug ................................................................................................................... 132
12 Business Plan ........................................................................................................................................ 133
  12.1 Vision ........................................................................................................................................... 133
  12.2 Sales Volume Estimates ................................................................................................................ 134
  12.3 Fixed Production Costs ................................................................................................................ 134
  12.4 Variable Production Costs .......................................................................................................... 136
  12.5 Profitability ................................................................................................................................ 138

13 Conclusion ........................................................................................................................................ 140

14 Acknowledgements .......................................................................................................................... 141
  14.1 Calvin Community ...................................................................................................................... 141
  14.2 Others ........................................................................................................................................ 141
  14.2 Industrial .................................................................................................................................... 141

15 References ........................................................................................................................................ 142
# Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Traditional Theatrical Spotlight</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Subsystem Cost Breakdown</td>
<td>13</td>
</tr>
<tr>
<td>3</td>
<td>Spending Breakdown by Subsystem</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>System Location Diagram</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>Subsystem Block Diagram</td>
<td>25</td>
</tr>
<tr>
<td>6</td>
<td>Ideal Lighting Positions</td>
<td>27</td>
</tr>
<tr>
<td>7</td>
<td>Overall Lighting Subsystem Model</td>
<td>39</td>
</tr>
<tr>
<td>8</td>
<td>Thermal System Model</td>
<td>40</td>
</tr>
<tr>
<td>9</td>
<td>Light Ray Diagram</td>
<td>40</td>
</tr>
<tr>
<td>10</td>
<td>Electrical Component Block Diagram</td>
<td>41</td>
</tr>
<tr>
<td>11</td>
<td>Bridgelux Vero 29 LED Array</td>
<td>46</td>
</tr>
<tr>
<td>12</td>
<td>LED Driver Output Board</td>
<td>48</td>
</tr>
<tr>
<td>13</td>
<td>Dimming Control Board</td>
<td>48</td>
</tr>
<tr>
<td>14</td>
<td>Temperature Sensor Board</td>
<td>59</td>
</tr>
<tr>
<td>15</td>
<td>Spotlight with Stand</td>
<td>54</td>
</tr>
<tr>
<td>16</td>
<td>Parabolic Reflector</td>
<td>55</td>
</tr>
<tr>
<td>17</td>
<td>3D Printed Light Attachment</td>
<td>57</td>
</tr>
<tr>
<td>18</td>
<td>Lighting Software Block Diagram</td>
<td>58</td>
</tr>
<tr>
<td>19</td>
<td>Temperature Sensor Testing Output</td>
<td>60</td>
</tr>
<tr>
<td>20</td>
<td>Resistance vs. Step Setting</td>
<td>63</td>
</tr>
<tr>
<td>21</td>
<td>Raspberry Pi Model B, with 16GB SD Card Inserted</td>
<td>69</td>
</tr>
<tr>
<td>22</td>
<td>Modified Logitech Webcom</td>
<td>70</td>
</tr>
<tr>
<td>23</td>
<td>Video Pipeline Test with Raspberry Pi using Avconv</td>
<td>71</td>
</tr>
<tr>
<td>24</td>
<td>Video Pipeline Test with Raspberry Pi using Gstreamer</td>
<td>72</td>
</tr>
</tbody>
</table>
Figure 51: Tilt Motor and Pulley Assembly ................................................................. 105
Figure 52: Pan Motor and Pulley Assembly ............................................................... 106
Figure 53: Bearing Support Drawing ........................................................................... 106
Figure 54: Mirror Support Drawing ............................................................................. 107
Figure 55: Autodesk Inventor CAD Model ................................................................. 108
Figure 56: Complete Actuation System ....................................................................... 108
Figure 57: Labeled Spotlight System .......................................................................... 109
Figure 58: Found BLDC Motor ................................................................................... 111
Figure 59: Tilt DC Gearmotor ................................................................................... 111
Figure 60: Final Microcontroller Decision Matrix ....................................................... 116
Figure 61: Tilt Reflectance Calibration Sensor Assembly ........................................... 117
Figure 62: H-bridge Motor Controller Generalized Schematic ................................... 118
Figure 63: BLDC Motor Driver on Breakout Board ..................................................... 119
Figure 64: Pan Control Board Layout ......................................................................... 121
Figure 65: Tilt Control Board Layout ......................................................................... 121
Figure 66: Pan Control Board .................................................................................... 122
Figure 67: Tilt Control Board ..................................................................................... 122
Figure 68: Actuation Tilt Program Flow ..................................................................... 123
Figure 69: Actuation Pan Program Flow ..................................................................... 124
Table of Tables

Table 1: Project Milestones ........................................................................................................................................... 9
Table 2: Prototype Estimated and Actual Budget ...................................................................................................... 12
Table 3: Team Latency Budget.................................................................................................................................... 29
Table 4: Message Protocol .......................................................................................................................................... 35
Table 5: LED Design Alternatives .......................................................................................................................... 45
Table 6: LED Driver Alternatives ............................................................................................................................. 47
Table 7: Cooling Design Alternatives ........................................................................................................................ 51
Table 8: Controller Design Alternatives ................................................................................................................... 53
Table 9: Pins Defined in Software ............................................................................................................................. 53
Table 10: Spotlight Illuminance Test Results ........................................................................................................ 62
Table 11: Actuation Microcontroller Options ........................................................................................................ 113
Table 12: Microcontroller Development Time ....................................................................................................... 115
Table 13: Encoder to Control Mapping ..................................................................................................................... 114
Table 14: Fixed Costs .................................................................................................................................................. 116
Table 15: Variable Costs.............................................................................................................................................. 121
1 Introduction

As part of the Calvin College Senior Design course, Team 9 created an LED spotlight capable of robotically tracking a stage actor without human intervention. This document serves as an in-depth look at the team’s design process, final design, and prototype testing. The Context section of this chapter outlines the motivation for the project. The Proposal section introduces the project the team proposed as well as the Senior Design course. The Team Members section introduces team members by providing information about each member. The design norms section introduces the normative design principles the team considered throughout their design. Lastly, the Report Structure section details the remaining chapters in the report.

1.1 Context

Many different events use spotlights including theater productions, concerts, and some sporting events. Most current spotlights, like the spotlight shown in Figure 1, are large, power-hungry devices that require at least one dedicated operator to continuously point the spotlight whenever it is needed. During blackout, the operator may not be able to see the stage and thus may need to search the stage to reacquire the actor being followed. Even if the operator can see the actor there may still be an error between where the light is when it is turned on and where the actor is. These light corrections can distract an audience and detract from the professional appearance desired by the lighting designer.

![Figure 1: Traditional Theatrical Spotlight](image)

Some alternatives are available in the form of intelligent lighting fixtures such as the Elation Platinum series or Philips Vari-Lite series lighting fixtures. Intelligent lighting is lighting that can be pointed and controlled from the lighting board without direct manual operation. This allows the lighting to be
programed to follow the lighting scheme a show automatically. Intelligent lighting does not however automatically following a target, so followspots still need to be manually operated. A few spotlights have been made that do track automatically including the Wybron Autopilot II, but have not gained widespread adoption.4

1.2 Proposal

In order to solve the problems described above, the team proposed building a power-efficient, autonomously actuated spotlight. The spotlight would utilize high power LED technology to reduce power consumption. The spotlight would autonomously follow a person on stage during well-lit and blackout conditions and integrate with a majority of existing theater infrastructure.

The team presents this design as part of the Calvin College Senior Design course. The Calvin College Senior Design project is a two-semester course taken by every engineering student. In the course, students propose, design, and build a major project. The project involves both creating a design and justifying its feasibility with a business plan. Students work in teams of 3 to 4. As part of the course, students deliver a Project Proposal and Feasibility Study in fall semester, and a complete Design Report in spring semester. At the end of the course, students present their work to the public at the Senior Design Open House.

1.3 Team Members

The project team consists of the four members described below.

1.3.1 Phillip Fynan

Phillip Fynan is a senior Engineering student in the Electrical and Computer concentration. Phillips is from Santa Rosa, CA, and has worked as a research intern with Carnegie Mellon University. Outside of class, he plays Baroque and Renaissance recorder. Following graduation he will be attending Carnegie Mellon University.

1.3.2 Jake Roorda

Jake Roorda is a senior Engineering student in the Electrical and Computer concentration from Grand Rapids, MI. He will also graduate with a minor in Mathematics. He has worked as an engineering intern at GE Aviation System in Grand Rapids, MI and will be part of their Edison Engineering Program after graduation. During the school year Jake works in the repair shop at Meyer Music. Outside of work and school, he also plays clarinet in the Calvin College Wind Ensemble.
1.3.3 Lucas tenBrink

Lucas tenBrink is a senior Engineering student in the Electrical and Computer concentration from Holland, MI. He will also graduate with minors in Mathematics and Business. He has worked as a facilities engineering intern at Meijer Headquarters and as a manufacturing assistant at Rapid-Line. On campus Lucas also works at the Hekman Library and as an Engineering grader. Upon graduation, he will work as a control systems engineer at Dematic.

1.3.4 Khoi M. Vu

Khoi M. Vu (Kevin) is a senior Engineering student in the Electrical and Computer concentration from Hanoi, Vietnam where he attended high school. He has worked as a research assistant for Calvin College Professor Yoon Kim. Outside of class his interests include photography and hip-hop choreography.

1.4 Design Norms

Directly applying Christian ethical principles to engineering design can be a difficult process. Often it is not clear how the two should be related at all. The concept of design norms was created to help apply Christian ideals to the design process. The eight design norms identified in the Calvin College engineering program are Cultural Appropriateness, Transparency, Stewardship, Integrity, Justice, Caring, Trust, and Humility.

While every design norm can and should be considered for this project the team found it helpful to focus on three design norms that they found specifically applicable to their project. These were the design norms of Stewardship, Trust, and Humility. Stewardship deals with using resources well whether these be the resources of creation, team member time, or of Calvin College. Trust gets at the idea that the design should be dependable and that it should be presented honestly such that the user has reasonable expectations of the design’s capabilities. Finally, the design norm of Humility recognizes that humans are fallen and fallible and that anything the team would produce has the possibility to break or have errors that must be anticipated.

1.5 Report Structure

The remainder of the Design Report covers the design and implementation of the autonomous tracking spotlight system from a number of different angles. Chapter two covers the organization and management of the team. Chapter three outlines system requirements and deliverables. These include requirements needed for the satisfaction and safety of the end user as well as those detailed in the Senior Design course.
Chapters four and following cover the design of the system. Chapter four deals with the overall system architecture and global design decisions. Chapter five covers the communication between the subsystems. Chapters six through ten cover the individual components of the design. Chapter six covers the LED spotlight system. This system provides functionality similar to current spotlights for the tracking spotlight system. Chapter seven covers the sensing system. This system is used to sense where the actors are on the stage. The tracking system is covered in chapter eight. It is used to identify the position of the actor using the data from the sensing system. Chapter nine covers the user interface that will be used by the lighting technician to control the system. The actuation system that moves the beam of light to follow the actor is covered in chapter ten.

All of these systems are integrated to produce the final product, so chapter eleven covers system integration. Chapter twelve comprises a hypothetical business plan. It breaks down the fixed and variable costs of the final production design and the potential market for the system. Chapter thirteen concludes the Design Report with an overall assessment of the system and a look at possible future work, and chapter thirteen contains the team’s acknowledgements.
2 Project Management

Team projects require planning and accountability to ensure on-time completion and quality deliverables. The chapter addresses the need to plan ahead and provide structure to how the team operates. The Team Management section outlines the guidelines for communication, accountability, and conflicts. The Schedule section covers the Work Breakdown Schedule (WBS), which outlines a proposed schedule for the semester including work hours and a flow chart to show critical paths. The Division of Labor section shows how the overall system and duties are divided amongst team members. Lastly, the Operational Budget section outlines the preliminary budget and where funding is acquired.

2.1 Team Management

The team includes all those involved with the project. The most immediate persons involved in the project include the student members assigned to Team 9. These are Phil Fynan, Jake Roorda, Lucas tenBrink, and Kevin Vu. The student members are responsible for completing deliverables, conducting research, testing, and fabrication. Beyond the student members, there are also others who are involved with the project. These include the team advisor, industrial consultant, and any sponsors. The team advisor is Professor Steve VanderLeest. The advisor oversees and provides feedback on the progress of the team including the deliverables and team management. The industrial consultant is Tim Theriault from GE Aviation Systems. He is responsible for providing advice in regards to project management, design feasibility, and guidance with design challenges. The engineering lab manager is Bob DeKraker. He is responsible for part ordering and the maintenance of the engineering labs. The CEAC reviewers are Eric Walstra, Paul Rozeboom, and Mark Michmerhuizen. The reviewers are responsible for reviewing the overall project and providing the feedback to the team members. Identifying non-student team members involved with the project is important in regards to acknowledgment, because honesty and gratitude are integral parts in conducting a project.

Communication within the team is achieved through multiple methods. First, the team discusses immediate issues and reminders during the Senior Design class period. Next, phone numbers and emails are distributed, so that members can contact one another should the need arise. These numbers are available both visually in the project area (in this case, on the whiteboard) and accessible through mobile methods, such as a phone’s contact list. Initial deliverable meetings shall be scheduled more than three days before the deadline. This is to allow for adequate time to adjust work after the first meeting and facilitate a second follow-up meeting. For example, verbal presentations require the group to meet to create the presentation, confirm material with the group members, and practice giving the presentation. Then, a second meeting is held to practice again and confirm any edits or additions made to the
presentation. Team members also agreed to send email communication no less than a day before it is required for the message to be received and team members shall check emails twice daily to ensure those messages are read before the needed time.

Beyond communication, accountability shall be ensured in regards to work done by individual team members by following agreed upon rules. Every week, status reports detailing accomplished work and plans for the following week shall be completed with input from each team member. Accomplishments are described with a specific document reference or findings, not with vague descriptions such as, “I conducted research.” When each team member has included their information, each member shall review the document before being emailed to the advisor and copied to the team. If milestones are missed in regards to weekly status, other members must be informed if unforeseen issues are causing the project to take longer than expected to correct the delays. This requirement was unfortunately relaxed by the team for much of the beginning of second semester leading to delays in many milestones.

The team also produced a time log to track how many hours each person put into the project and how those hours were spent. Each instance of work performed by an individual team member is entered with their name and the date, hours, work category, and work description. Each member records total weekly hours in the weekly status report with the group total determined afterwards. The team shared documentation of their work in a Google Drive folder accessible to all team members and the advisor, so that team members could view each other’s work and the advisor could verify claims made in the status report. By keeping the documents in one place, team members also had the ability to prevent redundant work from occurring and prevent work from being lost in other storage locations.

The flowchart created for the Work Breakdown Schedule (WBS) shall guide work progress. Weekly status goals shall reflect dates and goals outlined in the WBS to make sure that delays do not occur from dependencies not being met. Lastly, challenges in meeting goals stated in the weekly status or WBS shall be communicated to other team members through meetings, phone calls, or email.

In regards to dealing with conflicts, rules for conduct and dealing with infractions were established. Should conflicts arise during the senior design project, the team shall communicate to the offending member or members that a problem has occurred and to try to work out a solution. For example, if a team member has been missing goals in the weekly status documents, the rest of the team is responsible for addressing the issue with the offending team member to better understand the situation and provide encouragement to fix the issue. As another example, if two team members are in a disagreement, other team members are responsible to mediate the disagreement to ensure dialogue can begin that will resolve the issue.
If direct communication between team members does not work, then the advisor shall be notified to prevent the conflict from escalating. In cases of split opinions in areas such as project decisions, professional conduct is to be expected. All team members are to both listen to and provide well thought out opinions. Team members shall also seek to be accommodating and compromising within a reasonable amount of time so that significant delays in the project are not created. For example, if a team member is having difficulties meeting a given deadlines, other team members shall be willing to aid the particular team member, so the task can be completed on time. The intent of compromise and dialogue is to prevent resentment between members that could adversely affect project progress.

Overall these methods for resolving conflict were effective. The team was in good terms with each other throughout the project largely as a result of each team member actively trying to avoid conflict. The only issue the team ran into was that they were generally overly adverse to conflict such that team members were not confronted when they failed to complete their required work.

2.2 Division of Labor

Ensuring quality work while managing the tight demand of team member work hours requires dividing up labor. The team accomplished this by dividing the overall system into separate subsystems with individual team members taking responsibility for a particular subsystem. For the spotlight project, the overall system can be dividing into subsystems including the lighting, actuation, sensing, user interface, tracking, and control systems. The team assignments for the project are as follows: Jake is responsible for the actuation system, Lucas is responsible for the lighting system, Kevin is responsible for the sensing system and user interface, and Phil is responsible for the tracking and control system. System integration is the responsibility of all members. This ensures that each team member considers system integration in their design.

The team chose Jake to lead the actuation system because of his familiarity working with motors and robotic devices. The team chose Lucas to lead the lighting system because of his analog design experience. Similarly the team chose Kevin to lead the sensing system and user interface systems because of his previous research into video streaming from embedded platforms, and they chose Phil to lead the tracking system based on his research into visual tracking algorithms and his substantial software experience.

These divisions are intentionally flexible. As an example, Jake took on the task of building a prototype beacon for early testing. This made sense because Kevin was busy looking for sensors and an early prototype of the sensing system with a camera and beacon was needed prior to a prototype of the
actuation system. Similarly, Phil has helped with the communications to the actuation system and from the sensing system because of his experience with Ethernet communication. At the start and midpoint of the second semester, the team also looked broadly at the division of labor and group member responsibilities and make any changes they felt necessary. For example, networking in regards to Ethernet communication was shared with Jake when tracking concerns took longer than expected. Implementation responsibilities are shared and require multiple team members to work together to ensure proper function. For example, the spotlight fabrication for housing the actuation and lighting subsystems required cooperation of the team members involved. Also, communication specifications between the subsystems were worked out in a meeting so that the information sent between systems was standard, understood by each member, and sufficient for operating within each subsystems requirements. Despite this flexibility, some deadlines still slipped because a given task could only be completed by a single person with the knowledge of the system or because moving the responsibility would significantly affect the evaluation of the team member who was behind.

Responsibilities beyond the subsystems are also assigned to team members. Phil is responsible for team meeting setups and reminders. This is to ensure deliverables are completed on time and that all team members are aware of the meeting times and locations. Weekly status report document creation is conducted by Jake. This is to ensure that weekly status reports are created each week and finished on time. Milestone notification is to be conducted by Lucas. This is to ensure that the team does not forget deliverable due dates. Lastly, Kevin is responsible for notifying the advisor about conflicts in meeting deadlines. This is to ensure proper communication with the advisor in regards to missed goals or delay. These roles were reevaluated at the start of the second semester, but no changes were made as they had functioned well in the previous semester.

2.3 Schedule

Because of dependencies within the project, components of the project shall be scheduled ahead of time in order to prevent delays or conflicts as the project work progresses. For example, the circuit testing shall occur after circuit fabrication. Unpredictability shall also be addressed in the schedule. There is always some uncertainty in the amount of time available to any given team member and the time to complete a given task. Therefore, the schedule shall allow sufficient time to complete each task with the knowledge that some tasks may take longer than expected or that team members may be overwhelmed with responsibilities outside Senior Design. Lastly, deadlines shall be taken into account. There is little flexibility in document deadlines as well as the completion deadline, so the schedule needs to meet those deadlines.
The team created a work breakdown schedule (WBS) to fulfill the need for a schedule. The work breakdown schedule in Microsoft Project format can be downloaded from: http://www.calvin.edu/academic/engineering/2013-14-team9/links/Team_09_Scheduled_WBS.mpp . The WBS includes each subsystem and course deliverable and divides them into tasks less than eight hours in length. By keeping tasks short the project can be tracked with more granularity. Task completion hours and duration are intentionally overestimated to compensate for potential unforeseen issues. The overestimations range from 50-100% of the expected time with no issues assumed. Some extra buffer time is also allowed to minimize the potential of a single issue delaying a substantial portion of the project. From the length in days specified for each task, a flow chart was created with dependencies assigned for each task. This chart allows for dependencies to be compared with deliverable due dates to ensure that the dates can be met. In the end, the hours and duration estimates, built in deadlines, and prerequisite planning will allow the project to progress at a steady rate and be completed on time.

The critical path of the flow chart identifies the date the project can be completed based on work predictions and dependencies and also identifies areas where delays may be the most costly. The critical path shown in the flow-chart arrives a few weeks before the deadline in terms of prototype work which means the team should be able to finish on time if the schedule is followed. A built in contingency plan encompasses the period from Christmas break through Interim as it has no scheduled tasks to be completed, but work is expected to continue. This period of time will allow for time to complete tasks that should have been completed before the end of the semester yet can be completed before the deadlines arrive after Interim. Lastly, milestones in the project were identified. These milestones with the accompanying dates and associated work to be completed are included in Table 1. This helps provide and a frame of reference for determining dependencies and critical path elements.

<table>
<thead>
<tr>
<th>Milestone</th>
<th>Completion Requirements</th>
<th>Due Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPFS Draft</td>
<td>• Subsystem Research</td>
<td>November 11, 2013</td>
</tr>
<tr>
<td>Revised PPFS</td>
<td>• Initial Design Work</td>
<td>December 9, 2013</td>
</tr>
<tr>
<td>Preliminary Budget</td>
<td>• Component Research</td>
<td>December 9, 2013</td>
</tr>
<tr>
<td>Safety Inspection</td>
<td>• Preceding Safety Self-inspection</td>
<td>March 5, 2014</td>
</tr>
<tr>
<td>Design Report Draft</td>
<td>• Testing</td>
<td>April 25, 2014</td>
</tr>
<tr>
<td></td>
<td>• Detailed Circuit Designs</td>
<td></td>
</tr>
<tr>
<td>Project Demonstration</td>
<td>• Functional Prototype</td>
<td>May 10, 2014</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Final Design Report</td>
<td>• Prototype Setup</td>
<td>May 14, 2014</td>
</tr>
<tr>
<td></td>
<td>• Course Conclusion</td>
<td></td>
</tr>
</tbody>
</table>

An estimate for the total number of project work hours, as summed from the components in the WBS, is approximately 365 hours or approximately 90 hours per team member. This works out to about five hours a week per team member which is well within the capabilities of each team member to accomplish.

At the end of the first semester the team had spent 307 hours on the project. At this point the team realized that the total hours was a gross underestimate. The team is currently burning through hours at roughly twice the estimated rate. This entails that the previous estimate likely had a 100% error. This is largely due to the greater than expected time cost of early research and the PPFS report. The team expects this trend to continue. To reflect this, the team has revised the total hours estimate to 730 hours or approximately 180 hours per team member. At ten hours a week this is a more substantial time commitment, but it is one that the team has showed an ability to meet over the course of the first semester.

As the second semester, the team had logged more than 1100 total hours. This is again larger than the estimate made prior to the start of the semester, but it is closer to that estimate with a margin of error or 65%. The tasks that took the most time during the second semester were the production of the physical lighting and actuation components, and the design of the communications software. A lesson that the team will take from this project is the need to break down to project to small pieces in the work breakdown schedule, as they found that tasks that seemed simple were actually quite involved when it came time to complete them.

### 2.4 Operational Budget

The primary source of funding for the project is the Calvin College Engineering department. The projected department funding for each Senior Design group was estimated at approximately $500, so to be good stewards of the funding provided it is important for the team to operate within the confines of a detailed and realistic budget. The project budget is set by the department in response to a budget request near the end of the first semester.

The primary cost driver for the project prototype is the required hardware costs. These costs make up the over 75% of the budget because of the quantity and variety of components needed to construct the four distinct subsystems that together will form the automated spotlight system.
The system’s complexity also leads to many component selection design. Many of these decisions involve balancing added features and functionality against cost and design time. Because of this flexibility, the team was careful to make a budget that provides a reasonable amount of flexibility in the system design and parts selection along with an allowance for unexpected expenses. The team also took into account the need to use Calvin College’s funding wisely, and thus made sure that cost effectiveness was part of every major design decision. After the initial budget requests were completed, the course administrators went back to the teams to ask if there were any cuts that could be made to the budgets in order to bring the class total down to the budgeted value.

The team felt that it was important to cut what they could from the budget. The team’s shared Christian faith led to this decision on account of the biblical virtue of humility and the biblical call to stewardship. In this instance, the team is called to be good stewards of money that belongs to the Calvin College Engineering Department. Being good stewards of this money not only relates to not squandering the funds, but it also entails using the funds to give the best possible returns for their owner the Engineering Department. In this case, that does not necessarily mean a monetary return, but rather a return in the form of educational value. As the team looked at budget items, they considered the benefit those funds would have to the team and weighed that against the benefit they may have to another team.

When the team humbly looked at the budget, they found that some items could be cut without significant detriment to the project. These areas were most often related to stretch goals that did not make up the core of the project such as adjustable focus or a lighting board control. The team was able to make other cuts on account of progress with prototypes and testing that eliminated some risk from particular sections such as the infrared camera and the sensor electronics. With the secondary revision, the initial budget request of $920 was revised to the current budget request of $775. This results in a requested budget reduction of 16%.

The team continued to keep their budget in mind as the project continued. During component selection, part cost was considered as a design criterion for any components that needed to be purchased. Also, when parts were ordered, samples were requested to attempt to reduce cost when applicable. Companies, such as Texas Instruments, often offer samples of parts, like LED or motor drivers, for educational purposes. Some costs allocated in the original budget were revealed to be inadequate for their specified purpose. This meant that some designs were changed. For example, the optics design was changed from an elliptical design to a parabolic and custom stray light filter design. The lenses required in the original elliptical design were found to be around a hundred dollars each. The design was changed so that the reflector handled the focusing with the resulting stray light handled by custom made parts.
The projected prototype budget is shown in Table 2. It shows both the true and estimated amounts. Figure 2 shows the cost distribution by subsystem as a pie chart, and Figure 3 shows the actual distribution of funds between the various project components. Note that although the actuation system parts cost more than expected, this could be covered through a reallocation of funds from the sensing and control budget. The final budget numbers show that the team was only $5.77, or 0.8% over budget.

Table 2: Prototype Estimated and Actual Budget

<table>
<thead>
<tr>
<th>VENDOR</th>
<th>ITEM</th>
<th>Est. Cost</th>
<th>Actual Cost</th>
<th>% Under (Over) Budget</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actuation Mechanical</td>
<td></td>
<td>$ 135.00</td>
<td>$ 195.33</td>
<td>-45%</td>
</tr>
<tr>
<td>Amazon</td>
<td>Large Bearing</td>
<td>$ 135.00</td>
<td>$ 195.33</td>
<td>-45%</td>
</tr>
<tr>
<td>Grainger</td>
<td>V Belts and Pulleys, Mach. Screws</td>
<td>$ 135.00</td>
<td>$ 155.33</td>
<td>-109%</td>
</tr>
<tr>
<td>SDP-SI</td>
<td>V Belts and Pulleys</td>
<td>$ 135.00</td>
<td>$ 166.00</td>
<td>-23%</td>
</tr>
<tr>
<td>ServoCity</td>
<td>Gearing, Bearings, Shafting, Shaft Collars</td>
<td>$ 135.00</td>
<td>$ 166.00</td>
<td>-23%</td>
</tr>
<tr>
<td>Actuation Electrical</td>
<td></td>
<td>$ 100.00</td>
<td>$ 208.78</td>
<td>-109%</td>
</tr>
<tr>
<td>Amazon</td>
<td>Cable and Connectors (Pointing Electrical)</td>
<td>$ 21.90</td>
<td>$ 29.00</td>
<td>-32%</td>
</tr>
<tr>
<td>Digikey</td>
<td>E-Stop Switch, Current Sense Resistor</td>
<td>$ 40.33</td>
<td>$ 47.68</td>
<td>-18%</td>
</tr>
<tr>
<td>Pololu</td>
<td>Voltage Regulators, Cabling, Motor, Mounting Bracket</td>
<td>$ 129.77</td>
<td>$ 149.82</td>
<td>-15%</td>
</tr>
<tr>
<td>Sparkfun</td>
<td>Cable and Connectors (Pointing Electrical)</td>
<td>$ 16.58</td>
<td>$ 27.00</td>
<td>-20%</td>
</tr>
<tr>
<td>Spotlight Electrical</td>
<td></td>
<td>$ 100.00</td>
<td>$ 141.82</td>
<td>-42%</td>
</tr>
<tr>
<td>Avnet Express</td>
<td>LED Power Module</td>
<td>$ 83.75</td>
<td>$ 111.08</td>
<td>-29%</td>
</tr>
<tr>
<td>Digikey</td>
<td>LED Array</td>
<td>$ 46.44</td>
<td>$ 51.25</td>
<td>-9%</td>
</tr>
<tr>
<td>Mouser</td>
<td>Digital Potentiometer</td>
<td>$ 11.63</td>
<td>$ 13.50</td>
<td>-15%</td>
</tr>
<tr>
<td>Spotlight Mechanical</td>
<td></td>
<td>$ 220.00</td>
<td>$ 193.17</td>
<td>12%</td>
</tr>
<tr>
<td>Calvin</td>
<td>3D Printed Light Collimator</td>
<td>$ 111.08</td>
<td>$ 111.08</td>
<td>0%</td>
</tr>
<tr>
<td>Digikey</td>
<td>LED Heat Management</td>
<td>$ 61.99</td>
<td>$ 67.66</td>
<td>-9%</td>
</tr>
<tr>
<td>Mouser</td>
<td>LED Reflector and Cable</td>
<td>$ 20.10</td>
<td>$ 21.00</td>
<td>-4%</td>
</tr>
<tr>
<td>Sensors</td>
<td></td>
<td>$ 145.00</td>
<td>$ 41.67</td>
<td>71%</td>
</tr>
<tr>
<td>Amazon</td>
<td>Visible Light Webcam</td>
<td>$ 30.79</td>
<td>$ 30.79</td>
<td>0%</td>
</tr>
<tr>
<td>Digikey</td>
<td>IR LEDs for Beacon</td>
<td>$ 10.88</td>
<td>$ 10.88</td>
<td>0%</td>
</tr>
<tr>
<td>Main Control Unit</td>
<td></td>
<td>$ 75.00</td>
<td>$ -</td>
<td>100%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>$ 775.00</td>
<td>$ 780.77</td>
<td>-1%</td>
</tr>
</tbody>
</table>
Figure 2: Subsystem Cost Breakdown

Figure 3: Spending Breakdown by Subsystem
3 Objectives and Requirements

The objectives and requirements chapter outlines system requirements and project deliverables. The requirements are divided by user interface, installation, communication, safety, power, and performance. The deliverables cover reports, designs, and other documents that are required by the syllabus or to support the prototype. User interface requirements encompass the requirements of the GUI the system user will interact with. Installation requirements encompass the requirements involved with setting up the system. Safety requirements include design requirements that are required to prevent injury or harm to a person. This is in accordance with the design norm of trust. Power requirements encompass the energy use specifications of the system. Lastly, performance requirements encompass the performance specifications that the system shall achieve.

3.1 User Interface Requirements

3.1.1 Centralized Control

Requirement: The control of the system shall be centralized to ensure the system is capable of being used by a single person.

Explanation: This is to increase simplicity and efficiency. Therefore, adjustments to the system, such as turning the light on and off, adjusting the brightness, selecting the subject, or directing the spotlight manually, shall be handled by the graphical user interface (GUI) so that the user is not required to make adjustments at other points in the system, wasting time in the process and possible requiring the help of another person.

3.1.2 Manual Direction Functionality

Requirement: Manual control shall be available to the user through the UI and allow the user to control the device directly without automatic tracking.

Explanation: One of the identified design norms for the project is humility. In regards to the project, it cannot be assumed that the tracking will work in all situations or that all situations can be covered by automatic tracking, so it shall be possible to disable it.

3.1.3 Simplified Control

Requirement: The user interface shall be a graphical user interface.

Explanation: The lighting system will be operated by lighting technicians with limited electrical engineering knowledge. The technicians will need a familiar interface, so the input shall be done through
simple GUI means such as buttons, slide adjusters, or keyboard control, as opposed to entering commands into a command line.

### 3.1.4 System Response to User Input

**Requirement:** Individual adjustments to primary light or tracking functions shall take the user no more than 3 seconds.

**Explanation:** Poor system response to input into the UI would reduce the utility of the system as changes would need to be planned as they could not be implemented rapidly without noticeable lag in the performance. In this case, adjustments shall include lighting intensity changes, tracking target changes, or manual light pointing changes.

### 3.1.5 Interaction Demand during Performances

**Requirement 1:** Under normal operation, tracking a single person with the spotlight shall demand no more than 10% of the operator’s time during a performance.

**Explanation:** Time counted towards the 10% shall include making adjustments with the GUI, making adjustments to the spotlight directly during the performance, or running beacons to performers if required. This does not include performance planning or system configuration.

**Requirement 2:** As a further constraint, the operator’s attention shall be required no more than a total of 30 seconds during any 5 minutes of operation.

**Explanation:** The 30 seconds shall only include adjustments made through the graphical user interface. The requirement can be tested by estimating how many adjustments need to be made for a specific task during a time period with a time estimate for each adjustment.

### 3.2 Installation Requirements

#### 3.2.1 Spotlight Mounting

**Requirement:** Spotlights shall be able to be mountable in a catwalk setting as is typical for many theaters.

**Explanation:** Catwalks include battens (pipes) on the edges for spotlights to be mounted. Also, a catwalk can support a stand or tripod of approximately less than 2 feet wide. For reference, the catwalk in the Calvin College Covenant Fine Arts Center (CFAC) is approximately 3 feet wide. The measurements of the spotlight for meeting this criteria shall be taken as the widest point of the system in the upright position.
3.2.2 Conspicuousness

**Requirement:** The spotlight shall not be required to be setup in a conspicuous position that distracts from performances.

**Explanation:** A conspicuous position is one that is in front of the audience’s view of the performance or directly adjacent to audience members seated in areas other than the sides of the auditorium or the balcony. In front of the audience’s view is defined as any part of the spotlight blocking view of any part of the stage of a seated audience member.

3.2.3 Flexibility

**Requirement:** The spotlight and camera shall be able to adjust positions with relative simplicity.

**Explanation:** This means that the adjustments to their mounting positions can either be made by hand (such as unscrewing a c-clamp) or by common tools such as screwdrivers.

3.3 Communications Requirements

3.3.1 Facility Data Transfer Cable Compatibility

**Requirement:** Long distance data transfer cables used by the system shall be compatible with communication cables typically used in theater settings such as Ethernet or DMX.

**Explanation:** This is because a system that requires installing new cables through an auditorium is expensive for potential customers. Ethernet and DMX are cables CTC and OCCE identified as the communication cables used in Calvin College’s auditoriums as well as in most other theater venues and both are typically run with commodity cat 5 cable at relatively low cost. Any communication distance between two devices that is over 9 feet is to be considered long distance.

3.4 Safety Requirements

To conform to the Christian design norm of trust, the system shall be designed with the safety of individuals involved in the system such as the audience, stage performers, lighting technicians, and any others who may work with the system.

3.4.1 Electrical

**Requirement:** The system shall comply with electrical safety practices.
Explanation: If potentially dangerous voltages or currents are used in a subsystem any conductive case materials shall be tied to earth ground and the lines shall be protected by appropriate fuses. The National Institute of Standards and Technology (NIST), an agency of the Chamber of Commerce, defines any RMS voltage between 120V and 1000V and current above 2 mA AC as moderate voltage which require mandatory safety fencing and warning signs if unattended and exposed. The team expanded this definition to include DC or RMS voltages over 40V at 2 mA or above.6

3.4.2 Mounted Components

Requirement: If mounted overhead, the spotlight and pointing system shall have a safety chain attached to the batten pipe.

Explanation: This shall be a secondary attachment to prevent the system from falling and injuring a person located beneath the spotlight.

3.4.3 Obstructions

Requirement: The moving parts of the actuation system shall be able to contact physical obstructions including other lighting apparatuses and people without damaging the actuation system or injuring the person.

Explanation: The system user shall be alerted to this issue and the system placed into a halted mode until the user can verify that it is safe to return to normal operating condition.

3.4.4 Materials

Requirement: Fabricated components for the system shall not have sharp or jagged edges which are capable of cutting or bruising individuals.

Explanation: For example, if an enclosure is made for the spotlight or electrical components from sheet metal or other base materials, the edges shall be filed, folded, or covered so individuals will not be at risk of cutting themselves when interacting with those components of the system.

3.5 Power Requirements

3.5.1 Energy Use

Requirement: The energy used by the spotlight shall be considered so that the design does not use more energy than is necessary.
Explanation: A design norm that the project shall conform to is stewardship which involves using resources efficiently and responsibly. To conform to this design norm, the system shall use less than 1KW, which is currently a typical wattage for theater followspots. For reference, the popular ETC Source Four uses 750W. Another popular spotlight, the Philips Selecon Pacific followspot has power settings at 575W, 750W and 1000W.

3.5.2 Power Source

Requirement: System shall be powered by easily available grid power from 15A, 115V wall outlets.

Explanation: This includes non-portable components such as the spotlight, control system hardware, and tracking camera.

3.6 Performance Requirements

3.6.1 Tracking Accuracy

Requirement: When moving, the spotlight position shall overshoot no more than 30 cm (approx. one foot) from the desired position.

Explanation: The desired position is measured as the vertical center of the subject. The overshoot is measured as the distance from the center of the beam to the center of the subject.

3.6.2 Lighting Conditions

Requirement: The system shall be able to operate under both fully lit and blackout conditions.

Explanation: Blackout condition means that all interior lights are off and no outside light is coming into the venue. Fully lit implies a setting where many of the theater’s spotlights are trained at the stage and architectural lights are turned on.

3.6.3 Tracking Subject Loss

Requirement: The spotlight tracking system shall operate despite loss of the beacon signal.

Explanation: An unexpected beacon signal loss could be caused by anything that block the infrared signal or visual edges of the subject at a 100% overlap lasting for more than 5 seconds.

3.6.4 Environmental Conditions

Requirement 1: The system shall function in temperatures between 40° and 100° F.

Requirement 2: The system shall tolerate relative humidity between 5% and 95%.
Explanation: This accounts for all expected indoor environmental conditions that could be encountered.

3.6.5 Actuation

Requirement 1: The spotlight shall be able to rotate at least 60° per second and tilt at least 30° per second.

Requirement 2: Spotlight shall be able to pan at least 180° and tilt at least 120° such that it can aim at any place on the main stage, stage calipers, and the front few rows of the auditorium.

Explanation: These requirements are needed to for the spotlight keep up with fast moving stage actors with the spotlight in all desired mounting positions. While this requirement will vary by installation location and venue, the team has determined through the study of various auditoriums that the ranges given above will be sufficient in the substantial majority of cases. The calculation of these figures is covered in the Actuation Subsystem chapter.

3.6.6 Beam Edge Quality

Requirement: The Spotlight shall have a clear beam edge on the stage.

Explanation: The average illuminance within the beam shall be no less than 60% of the illuminance at the center. Then the illuminance shall drop to below 10% of center illuminance within one foot from desired edge of the beam.

3.6.7 Beam Radius on Subject

Requirement: The beam shall have a radius of 5 ft with center referenced from the waist of the subject.

Explanation: This gives a radial margin of error of approximately 2 ft for a 6 ft individual.

3.6.8 Noise

Requirement: Spotlight shall be no louder than 30dBA at a distance of 10 feet when operating to go unnoticed during a production.

Explanation: This allows the spotlight to operate with the noise going unnoticed. For reference, this is no louder than a moderate whisper.\(^8\)

3.6.9 Illuminance on the Stage

Requirement: Maximum illuminance or light intensity of the spotlight beam at the subject on stage shall be greater than or equal to 50 footcandles and less than or equal to 100 footcandles.
Explanation: This is identified as typical “performance for visual task of medium contrast or small size” or “illuminance on task.”

3.6.10 Light Output

**Requirement:** The absolute minimum light output that the system at full output shall achieve is 4000 lumens.

**Explanation:** The maximum light output is determined using the minimum beam radius of 5ft and minimum illuminance of 50 footcandles. Footcandles is composed of lumens per square foot, so the minimum light output shall be \( \pi \times 5^2 \times 50 = 4000 \) lumens.

3.6.11 Color Temperature

**Requirement:** The color temperature of the light source shall be greater than or equal to 3000K and less than or equal to 3500K.

**Explanation:** This requirement is included so that the spotlight will conform to the color temperatures that are common for spotlights used in theater spotlights. OCCE identified this color temperature range as criteria for purchasing spotlights for Calvin College’s venues such as the CFAC auditorium. The reason given for this is that the specified range describes a warm, yellow-white light output, which is desirable when putting on a performance. Lower color temperatures are considered warmer (2700K) and higher color temperature (6500K) are considered cooler.

3.7 Deliverables

3.7.1 PPFS Report

The Project Proposal and Feasibility Study (PPFS) is a report that outlines proposed design, test, and implementation of the senior design project. Included in the PPFS are team organization, system requirements, subsystem research and design, and a business plan. The purpose of this content is to outline the project at a detailed enough to outline the project and prove that it is feasible.

3.7.2 Working Prototype

A working prototype is the proof of concept that demonstrates the how the system works, proves that it was feasible, and gives a means of testing the design to help foster improvements. Ultimately, a theoretical design is unable to prove the auto tracking spotlight concept as well as a working prototype can. The working prototype shall be demonstrated at Senior Design Night.
3.7.3 Final Report
While the PPFS gives an outline of the project and proves its feasibility, the final report outlines testing and debugging that took place, provides more detailed design work, and outlines how the prototype works.

3.7.4 Business Plan
The business plan outlines the estimated cost of the system in terms of component cost, costs of production, and overhead. Also, the market for the product will be defined and demand will be estimated. With cost and sales estimates in place, a pricing strategy is determined with the overall profit estimated at the end. For the purposes of the PPFS, the business plan is a simply a chapter of the report. The updated business plan is to be included in the final report.

3.7.5 Software
Along with the working prototype, a major component of the project is the software that runs the prototype. The software includes the UI implemented on the main controller and how it communicates with the subsystems as well as the software of the microcontrollers that run the subsystems. This software shall be published and provided with the final report.

3.7.6 Hardware Specifications
Hardware specifications including the circuit schematics and CAD drawings of constructed components are also to be submitted. These will offer detailed information about the system that was designed which cannot be completely covered in the PPFS or Final Report. Optionally, designs may be included that go beyond the prototype to better reflect the business plan and provide insight into how a market product would be designed beyond the budget and time constrained prototype. This would also reflect design and testing results or observations that present ideas for future versions of the design.

3.7.7 Design Documentation
Design documents detailing research or other details of subsystems that are not included in the final report are to be offered within the shared Google Drive. This allows for access to detailed progress and work to each team member and the team’s advisor.

3.7.8 Team Website
The team website offers an overview of the system for someone browsing Calvin Engineering senior design projects. The main page offers a brief explanation and outline of the project. A more detailed
explanation of the project and how it works is included on another page as well as descriptions of each of the team members. Important documents such as the PPFS and Final Report will be available upon completion.
4 System Architecture

This chapter describes design decisions that apply broadly to the entire tracking spotlight system. These decisions begin with the identification of the target customer and deciding upon the scope of the system. The breadth of this section is then devoted to the division of the project into subsystems, where to place certain functions, and how to connect the subsystems.

4.1 Design Criteria

4.1.1 Customer Identification

The team’s first step in designing this system was to identify the customer. The team met with both the Calvin Office of Campus Events (OCCE) and the Calvin Theatre Company (CTC). Each of these groups had their own unique system requirements and vision for the end product that are representative of a larger body of potential customers. OCCE envisioned a larger spotlight that could be used at their larger venues such as the Calvin College Covenant Fine Arts Center and the Van Noord Arena. Their main concern was an easy to use interface that could be operated live during a show.

Alternatively, CTC was looking for a much smaller spotlight that would fit inside the lighting areas inside their more compact venues including the Gezon Auditorium and the Lab Theatre, a contemporary black-box theatre. They were also primarily concerned with having a system that could interface with a DMX lighting board so that the light could be programmed into the scenes that make up a theatre production.

Both of these potential customers bring their own set of expectations to the system. OCCE is a good representation of what a touring musical group or amphitheater customer would desire from this system. Within the Grand Rapids area, the Van Andel Arena and the Frederick Meijer Gardens would be potential customers in this class. CTC is a good representation of the academic and community theatre market. The Grand Rapids Civic Theatre and the Forest Hills Fine Arts Center would be examples of this type of customer within the Grand Rapids area.

Whenever possible the team’s goal is to please both types of customer in order to broaden the potential customer base of their system. That said, the team also knows that one type of customer must take precedence for the design to be focused and not end up a victim of compromise. The team selected CTC and in general small theatre customers as their primary customers because their smaller venues would in general reduce the cost of the prototype system and make the sensing easier as well.
4.1.2 Scope Restrictions

The team recognized early on that this is an ambitious project. To attempt to control the cost and amount of work required by the project it is important to set up some restrictions on what will be attempted in the project prototype. The primary goal for the prototype system is to provide a proof of concept for the methods of sensing, tracking, actuation, and lighting. The prototype system did not attempt to fully match the performance that would be necessary for a saleable professional product. The largest restriction comes from the realization that the tracking system is the core of the project. As such, the team wants the prototype lighting system to be representative of a stage spotlight, but does not desire it to be a perfect replacement. Practically, this means that the team wants to reach a similar light output, but matching the exact lens structure and feature set is not a design goal. Another corollary is that while the team will design a finished end product, they will construct the prototype using available materials such as scrap metal and development boards that would not be feasible or cost effective for full run production.

4.2 Overview

To begin, the team had to make a high level conceptual plan that could meet the design requirements. They determined early on that a single entity system would not meet the requirements. While the team could integrate most of the electronics into the spotlight, control functions needed to be brought out to the lighting technician who is situated in the control booth. This necessitated a system made up of at least two components that communicate with each other. Figure 4 shows a block diagram of this design.

![Figure 4: System Location Diagram](image)

At this point the team conducted research into what communication method could be used to tie these systems together. The team identified a number of possible communications standards that could make this link. The standards included DMX, RS485 Serial, and Ethernet. The team left the final decision of communication protocols for later in the design process, but these viable options gave them the confidence that the system could be geographically separated successfully.
4.3 Subsystem Divisions

One of the first things the team had to determine was the overall design and segmentation of the system. Breaking down the overall system into parts allows the design work to be divided amongst the members of the team and it allows for more focused design work to continue on each component.

In this case, the automatic tracking spotlight system is most readily divided into five functionally and physically separate parts. These five subsystems are the sensing, user interface, tracking, actuation, and light subsystems. These systems are shown in the level one block diagram shown in Figure 5.

![Subsystem Block Diagram](image)

**Figure 5: Subsystem Block Diagram**

The sensing subsystem is comprised of the visible and near-infrared sensors that are used to detect the position of each person on the stage. This system uses a near infrared sensor to find the location of near infrared beacons worn by the actors. It will also include any additional hardware and software necessary to send the data from these sensors to the main control and tracking subsystem. The sensing system will stream the sensing data back to the tracking system which will perform the target identification and tracking calculations. This design decision is addressed further in section 4.3.1. The beacon is also part of the broader sensing system. The beacon system’s purpose is to mark the stage actors so that they can be detected by the infrared sensors on the sensing system. It consists of battery-powered infrared LEDs that are pulsed at a predetermined rate so they can be detected by the sensing system and identified by the
tracking system. This system will be small enough to be worn by the actor and inexpensive enough so that multiple beacons can be produced and used economically.

The tracking system is responsible for taking in the sensor data and using it to determine where the stage actors are in relation to the spotlight, so that the light can be pointed correctly. It outputs these actor positions to the control system in terms of camera pixel coordinates.

The user interface and control system allows the user to manually point the light and alter its brightness while also allowing the user to select what actor to track. It takes in the actor positions from the tracking system so they can be overlaid on the video feed from the sensing system. Based on user input from direct user control and from the optional DMX connection, the control system will send absolute positioning commands to the actuation system and lighting commands to the lighting system. At this level, the control system will be an open loop system. This means that the system will use a predetermined relation to correlate lighting actuation positions with where the light will fall on the stage. A closed loop system would use a system to sense where the light from the spotlight hits the stage and adjust for any errors. With other lights in use this could be very prone to interference, so the team chose to pursue an open loop solution.

The actuation subsystem is responsible for pointing the beam of light from the spotlight at the desired point on stage. It is comprised of both the hardware and motors necessary to aim the light and also the electronic components needed to communicate with the control and tracking subsystem and to drive the motors in a smooth and precise manner. It uses a closed loop control system such that its controller can determine the true location and speed of the moving components and correct for any errors. The design will use both absolute position sensing and differential sensing to obtain good feedback for an active system and maintain absolute positioning accuracy after startup.

Finally, the light subsystem is comprised of the digitally controlled spotlight itself. It receives commands from the control and tracking subsystem that determine the desired light intensity and possibly spot size. The goal of this system is to produce a spotlight with performance similar to other small spotlights with added efficiency and the ability for the light to function without direct human operator.

### 4.3.1 Subsystem Locations

The team decided on these divisions for a number of reasons. One of the most basic reasons was the need for different components to exist in different places.
Part of the appeal of a robotic spotlight system is the wide variety of possible mounting locations. Existing spotlights can only be mounted on large catwalks or alcoves to allow the operator space to work. After talking with representatives from the Calvin Theatre Company and the Calvin Office of Campus Events the team learned that these typical locations are often compromises. In these meetings the team learned that the typical textbook mounting location for a spotlight is 45° above and 45° to the side of the performer as shown in Figure 6. In a typical theatre this means two lights may be used to provide better coverage with one mounted 45° to the left and another mounted 45° to the right. 45° above vertical is ideal because it is low enough to place light on the front of the actor without creating large shadows, but also high enough so that it does not blind the actor and flatten their facial features. It has also been postulated that light between the angles of 30° and 60° looks most natural because that is the typical angle of sunlight. Spotlights are typically placed 45° to each side so that the entire front face of the actor is illuminated by two spotlights without shadows on either side. The team was also able to confirm these numbers in existing lighting literature.

When these ideal angles are translated into a position in the theatre, it usually will correlate to mounting the light on one of the first few catwalks in front of the stage. Early on, the team also decided that it would be advantageous to mount the sensing cameras as near to the spotlight as possible. If the spotlight and the cameras are separated by more than two feet, significant parallax between the two will result and the tracking system will need to track in three dimensions to correct for this. However if the cameras and
the spotlights are essentially in the same location the three dimensional problem can be reduced to a two dimensional problem that is simpler and less expensive to solve.

Despite being placed in the same location, it makes sense to fully separate the sensing and lighting units because the function of these units is quite different. Both streaming video and controlling motors and lights have unique hardware needs. The sensing system needs cameras and fast processors with hardware capable of encoding mid to high resolution video. The lighting and actuation systems only need a simple microcontroller with easily accessible discrete IO for lighting and motor control. Because there is little shared hardware, the cost savings generated by combining these systems would be small. Trying to integrate these systems would also make the design of both systems much harder because of the possible resource conflicts and dependencies.

While the sensing system could likely accommodate the extra computational load of the actuation system, the added design pressures of simple discrete IO would make the resulting microcontroller choice a compromise between these two applications. The extra development time that would result from using a single microcontroller was deemed by the team to be significant enough to justify the relatively small cost of one or two additional 8-bit microcontrollers.

In the end device, this separation also makes sense. The anticipation is that the sensing unit will be permanently installed in a given venue, but that the spotlight may be removed and stored out of the way when it is not needed. Based on past technological change, it will also likely be the case that the customer may desire to update the cameras in the sensing unit with newer more capable models long before the lighting system warrants replacement. Finally, for larger shows the lighting designer could opt to place two lighting units near one sensing unit in order to track two different people at one time.

The remaining subsystems are the tracking and control subsystems. In order to allow the lighting technicians to control the light some link needs to be made to the lighting booth. As justified in the tracking and control subsections the physical layer of these systems consists of a normal tower PC. The production design may move this computer to an electrical room in the form of a rack mount computer, but in general the system will remain the same.

4.4 Overall System Latency

The team determined that the maximum acceptable latency from the time the actor moves until the light moves to follow them is 500ms. This was calculated assuming the actor was moving at four feet per second with a spotlight beam that is two feet in radius wider than the actor. The time is then \( \frac{2 \text{ft}}{4 \text{ft/s}} \), or 500ms. If the actor were to suddenly stop the light would have 500ms to stop before leaving the actor
in the dark. The same is true for an actor that suddenly starts moving. 4ft per second was considered reasonable as actors and performers rarely run on stage, and if they do they usually accelerate into and out of the run gradually.\textsuperscript{14}

The team flowed the latency requirement down to the subsystems using the latency budget shown in Table 3. The largest single latency is that of moving the mirror. This makes sense because inertia limits how fast the motors and mirror can respond to a sudden change.

\begin{table}[h]
\centering
\caption{Team Latency Budget}
\begin{tabular}{|c|c|}
\hline
System & Latency Budget \\
\hline
Sensor & 50ms \\
\hline
Communication & 20ms \\
\hline
Tracking & 180ms \\
\hline
Actuation & 250ms \\
\hline
\end{tabular}
\end{table}

\section{4.5 Subsystem Responsibilities}

There are multiple functions that could be done in more than one subsystem, so before the team could begin designing these subsystems they had to decide on where these functions would reside.

\subsection{4.5.1 Tracking Computations}

The tracking computations could be completed by the sensing subsystem or by the control subsystem. Both of these systems would likely have computers, either a traditional desktop or a single board computer, running Linux.

The main advantage of doing the tracking at the sensing subsystem is that it eliminates the need to stream all of the sensing data to the control subsystem. The main advantage of doing the tracking at the control subsystem is that the control computer will likely be a full desktop PC in order to provide the user interface. This computer should have more processing power available for tracking than the embedded computer that is likely to be used for the sensing system. While the sensing system could be implemented with a larger computer, mounting such a large computer on a catwalk or lighting alcove would certainly be problematic. The team also reinforced this decision when they decided on using camera-based
tracking and providing the user with a view of the stage as part of the user interface. In this case, the camera feed would need to be streamed to the control unit regardless of where the tracking calculations were performed.

### 4.5.2 Motor Control

The motor control functions, likely PID loops, could be done by the actuation controller or the central control subsystem. The performance of these control loops is not a large issue as almost any microcontroller is capable of running a PID loop or similar control loop very well, so a more powerful computer is not a large advantage.

The main benefit of doing the control loop on the central control computer is the ability to quickly change the PID parameters. The main advantage of doing these calculations as part of the actuation controller is that there will be no communication delay in the control loop. The actuation system is required to respond in 250ms. This would not pose an issue if the PID loop only needed to run once to make a change, but it must run many times quite quickly in order to be effective.

In particular, the derivative term in the PID needs to run quickly to best approximate the instantaneous slope of the measured response. It is important that this term be faster than the natural frequencies of the system to prevent resonance oscillation. Without the physical servo system it is hard to know what their resonances will be, but it would not be unreasonable to expect frequencies up to 30 Hz. If the PID system is to correct for these oscillations it must sample fast enough to see them and to respond on the same part of the wave. This means the system needs to respond within ¼ to ⅛ waveform or in this case 4-8ms.

The team’s final decision was to place the control functions in the actuation controller to avoid the communications delays and so that this system can be developed and tested separately from the main controller.
5 Subsystem Communication

Each subsystem communicates with the others. The GUI communicates with the tracking system, which communicates with the actuation system which communicates with the light. The tracking system also communicates with the sensing system. While the requirements are similar, the different hardware of the different subsystems leads to different choice of communication protocol.

5.1 Subsystem Communication Requirements

The team began by identifying the requirements for the communications solution.

- The communications system shall be able to send positioning commands to the actuation and lighting system at an approximate rate of 320 bits per second.
- The communications system shall be able to stream 480p definition or higher video from the sensing to the tracking system at 30 frames per second or more.
- The communications system shall have a total latency less than 20ms as defined in the latency budget.
- The communications system shall be able to send data at least 200ft. This requirement is dictated by the distance from the control booth to the lighting catwalk.
- The communications system shall not be affected by interference common in performance venues. Interference that should be considered would include but are not limited to wireless microphones, cellular phones, Wi-Fi, parallel Ethernet traffic, and DMX lighting busses.
- The system shall provide a bandwidth of no less than 70.6 Mbps to carry video and commands. Video at 480p requires 70.3 Mbps of bandwidth. The video has 640x480 pixels with 1 channel and 8 bits per pixel at 30 fps, giving $640 \times 480 \times 8 \times 30 = 70.3$ Mbps.

5.2 Subsystem Communication Decision Criteria

Besides the requirements, other factors were considered when selecting a communications solution. First, the team looked for a single system that could meet both the actuation and sensing communications needs. While these could be separated into two different systems it would make the system simpler and more intuitive for the user if these could be combined into a single system.

The team also researched the cost of each option and also the amount of development time each alternative would need. The team also wanted to make use of existing wiring present in current theatres. From inspections of local theatres on and off Calvin’s campus the team found that the majority
of theatres were fully wired with Ethernet and DMX, and some theatres also had run undedicated category 5 cables that could be used for new DMX, RS485, or Ethernet runs.

While it would be possible to make new cable runs the team felt that this would conflict with their design norm of stewardship. The team believes strongly that they are called by God to be caretakers of this earth. With this in mind it does not make sense to run new cables if the existing cables could do the job. This also reduces the implementation cost of the system for the venue.

The team also considered the design norms trust and humility in the communications decision. Some options would only allow unidirectional communication. While this would work for normal operation it would not allow the actuation and lighting subsystems to report errors back to the control unit. While the team hopes the system will never fail, they are humble enough to know that the system will not be perfect and eventually it will fail. With a bidirectional communications system in place the actuation and lighting units can report error conditions so the operator is made aware of the situation. This also means the operator can trust the system a bit more because they know that no errors were detected.

5.3 Communications Alternatives

The team explored a number of different communication protocols. The team chose not to investigate wireless protocols in depth due to their higher cost and inherent susceptibility to interference. The interference risk is especially high in a concert environment filled with hundreds of cell phones and their associated Wi-Fi and cellular radios.¹⁵

The team turned its attention to wired protocols. USB, Firewire, HDMI, and RS232 were all eliminated because they could not transfer the data the required distance. The team in particular considered RS485, Ethernet, and DMX.

5.3.1 RS485

RS485 is a differential serial signal that is meant to be used for multi-drop networks with wiring distances up to 500 ft. This makes it ideal for use in a system like ours that could have many different lights scattered about the theatre. The primary difficulty in implementing RS485 comes from the lack of existing infrastructure. To use RS485 in a theatre one would have to run new wiring or repurpose existing CAT5 or CAT6 cable runs if they were available.¹⁶

5.3.2 Ethernet

Ethernet is the familiar network used to form the backbone of today’s computer networks. Like RS485, it is usually implemented with a differential signal.¹⁷ Ethernet communications are made up of packets that
are routed through the network to their assigned destination. Ethernet supports very high data rates, and its implementation is quite flexible. Today, many theatres have Ethernet drops throughout the building that feed into the main network.\textsuperscript{18} This existing infrastructure can be leveraged to greatly reduce installation costs. In situations where an Ethernet network is not in place customers have the option of using commonly available wireless access point technology to create a wireless link, though this could lead to the interference Ethernet was selected to avoid.

5.3.3 DMX

The third option that the team considered was the DMX standard. DMX is a digital multiplexed lighting communications protocol that is widely used to connect lighting boards and control computers to faders and DMX enabled hardware. The DMX standard assigns virtual channels to each light or controlled lighting option. Each channel carries one 8-bit value. If more resolution is desired for a given function two channels can be mapped so that they are combined to make a 16-bit channel. Up to 1024 virtual channels can be mapped to one DMX cable.\textsuperscript{19}

The DMX standard itself is built on top of the RS485 standard’s physical and electrical layer. It is run in most modern theatres and it is well understood by lighting technicians. There are three difficulties that come along with DMX. DMX assumes one directional communication. This is fine for stage lights, but it is not as desirable for the actuation and lighting systems as described previously. The channel setup is also quite restrictive when it comes to what data can be sent, so it would only be an option for the actuation system and not the sensing system. Lastly, while most theatres are wired for DMX, the DMX standard is relatively recent and some theatres still use other competing standards.\textsuperscript{20}

The team also looked at the possibility of sending arbitrary RS485 signals on a DMX bus. While this would be possible on an unused DMX bus, it would not be an option for a bus with active DMX communication, and it would thus not be an option in most theatres.

5.4 Communication Choice

The team picked communication options between each subsystem independently.

5.4.1 Tracking to Actuation and Camera

After examining the three standards in detail it became clear that the Ethernet was the best option. DMX was not chosen because it could only make up half of the communications system as it cannot stream video, and it cannot feedback potentially important status and failure data. RS485 was a bit more suitable, but it was eliminated because many theatres would have to be rewired to use it. The team also
determined that because it was new to the theatre industry it would likely cause a lot of confusion especially given its similarity to DMX. Finally, while RS485 is capable of streaming video there are not many implementations of this, so it would likely be up to the team to devise their own standard for this transmission. This would present significant challenges, so the team choose not to pursue this option. With Ethernet as the clear choice for video streaming, the design criteria favoring one communications standard instead of two lead the team to adopt Ethernet as the actuation and lighting communications system as well.

Once Ethernet was chosen, it added a needed function to the actuation subsystem. Because of the packet based nature of Ethernet communication the team could develop the video streaming independent of the spotlight commands. Because the spotlight commands are at a very low bit rate it is only necessary from a design standpoint to make sure that 0.2% of a 10Mbps Ethernet connection is reserved for the actuation commands.

5.4.2 GUI to Tracking

The team also picked Ethernet to communicate between the GUI and tracking systems, since Ethernet will reduce development time. Both systems are implemented on a general purpose computer which already has Ethernet hardware. The tracking system will also send video to the GUI system, so Ethernet is the clear choice.

5.4.3 Actuation to Lighting

Ethernet hardware, like the hardware for each of the design alternatives has some associated cost, so it does not make sense to have duplicate Ethernet hardware for both the actuation and lighting subsystems given that they are guaranteed to be within a few feet of each other when a low cost bus would do. The solution is to have the actuation subsystem contain the Ethernet hardware and have it pass along any messages to the lighting system over an appropriate short range serial bus. This adds only a marginal amount of work for the actuation processor, and it has the potential to reduce the overall system cost and implementation time.

The team assessed SPI, I2C, One-Wire, and Multiplexed Serial for use as this link. The original design called for the use of SPI to make this data link. The actuation controller would receive in commands from the Ethernet controller and pass it to the lighting controller over SPI. This would in theory work well, even though the actuation system is already talking to the Ethernet controller over the SPI bus, as the chip select line would indicate which slave device was intended. However, because the control of the chip
select line for the Ethernet driver is buried inside of thousands of lines of library code the team did not feel safe reusing that bus.

The final actuation microcontroller design also forced the choice of short range bus. This microcontroller only had one pin to spare to make this connection. This ruled out I2C as it would require 2 pins. This made it a two way decision between One-Wire and custom multiplex serial. This was the same decision encountered when connecting the two halves of the actuation system, see section 10.6, and thus the team chose to use multiplexed serial for the same reasons outlined in the actuation chapter. This made additional sense because this bus could leverage the proven code already written for the actuation system. In the final system, this bus was shown to be unreliable, so it was modified to only provide one-way communication. The team determined that this was acceptable as a lighting failure would be indicated by the extinguishing of an “80

5.5 Ethernet Implementation

The tracking system is the client and the light system is the server, i.e. the lighting system listens for the connection. The protocol is synchronous; the tracking system sends messages to the lighting system then waits for the response. The team identified the necessary messages for the protocol, shown in .

<table>
<thead>
<tr>
<th>Message</th>
<th>Message Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GOTO</td>
<td>‘G’</td>
<td>Points spotlight at given position</td>
</tr>
<tr>
<td>ONOFF</td>
<td>‘O’</td>
<td>Turns light on or off</td>
</tr>
<tr>
<td>HALT</td>
<td>‘H’</td>
<td>Halts motors</td>
</tr>
<tr>
<td>SLEEP</td>
<td>‘S’</td>
<td>Sleeps motor drivers</td>
</tr>
<tr>
<td>WAKE</td>
<td>‘W’</td>
<td>Wake motor drivers</td>
</tr>
<tr>
<td>INTENSITY</td>
<td>‘I’</td>
<td>Set light intensity</td>
</tr>
<tr>
<td>CALIBRATE</td>
<td>‘C’</td>
<td>Calibrate zero position</td>
</tr>
<tr>
<td>STATUS</td>
<td>‘T’</td>
<td>Get actuation and lighting status</td>
</tr>
</tbody>
</table>

The majority of the messages require a reliable transport. The Goto message, used to point the light, is the only exception, since it is sent frequently enough that it can be missed. Since reliable transport is necessary, the protocol was built on TCP, rather than UDP (which isn’t inherently reliable). UDP was considered, but building reliability on top of it would have increased development time.
The team considered using a RPC library to automate the protocol, to reduce development time. Both Protobuf and Thrift were investigated. Unfortunately, none of the libraries the team found would run on the microcontroller used in the actuation system. Instead, the team designed their own protocol. Since the requirements were simple enough, the development time wasn’t significantly increased.

The protocol is a simple binary protocol. A text-based protocol wasn’t used because it would require code to parse the numbers on the actuation system that would be quite time intensive. While the required bandwidth was also reduced by this choice, it was not as much of a concern as all messages were less than 8 bytes in total and the Ethernet receiver board will not receive less than 8 bytes of data.

Each message consists of a frame where the first byte is the remaining length of the frame. The next byte is the message type, and following bytes are the parameters for the message.

To test the protocol, the tracking system was connected to the actuation system with test data to verify the response is as expected.

5.5.1 GUI to Tracking System

The communication between the GUI and the tracking system is synchronous as well. Since the communication is between two general purpose computers, a RPC library was used instead of a custom protocol. Using a RPC library reduced development time and tied in with our design norm of humility. By using an existing solution, we are showing humility that we aren’t experts at writing communication protocols.

The team considered two RPC libraries: Protobuf and Thrift. Both are mature. However, Thrift doesn’t require any additional tools to do RPC, so it was chosen to reduce development time.

The messages sent over this communication system are the same messages that are sent between the tracking and actuation systems with the addition of a message to enable automatic tracking.

Both the GUI and the tracking systems were tested against small testing programs that only provided the correct interface before being tested together.
6 Lighting Subsystem

The lighting subsystem chapter provides an overview of the spotlight’s lighting subsystem which includes thermal, optic and electrical systems. The Spotlight Research section details the functions of spotlights in theater settings that the lighting system should reflect. The Lighting Subsystem Overview introduces the electrical, optical, thermal, and software systems of the lighting subsystem and how there are related to each other. The Lighting Subsystem Hardware section details the optical, thermal, electrical systems and accompanying assembly of the lighting subsystem and the design decisions made for their components. The Lighting Subsystem Software details the software design for the lighting subsystem microcontroller. Lastly, the Testing and Debug section outlines the process for testing and implementing the hardware as well as the debug procedure for the subsystem software.

6.1 Spotlight Research

While lighting design uses a broad spectrum of lighting devices, the particular spotlight outlined in this report is termed a followspot. Lighting design encompasses many shapes, sizes, and applications and as such uses a wide variety of devices. Some lights are completely automated and can be controlled through lighting boards, which program specific actions through queues. Others are wide and bright such as Fresnel and parabolic aluminized reflector lamps (PAR) to provide background light. But a followspot is defined by characteristics of manual control and focused light output. It is manual, because a figure on stage can move in an unpredictable manner where preprogramming through a lighting board is out of the question. Followspots are also narrow in their field as a particular person on stage is deemed the subject, not the entire accompanying scene.

Beyond the basic characteristics that define followspots, there are many characteristics that vary between followspots themselves. These can include color, brightness, focus, shape, and structure. Color can be changed with attachments called color gels, which attach at the end of spotlights or by the type of bulb that is used. Go-betweens that can be inserted into the lens tube, called gobos, can be used to change the shape of the beam. Tunable lenses and irises can be added to focus or reduce the diameter of the beam. However, many of these additions are add-ons to existing spotlights or only apply due to few models due to the added cost.

The majority of spotlights are purchased for warm, white light (3250K color temperature as defined by OCCE) and fixed beam angle. Warm temperature constraints have meant that incandescent and HID lamps are still used in most spotlights. Other bulbs, like fluorescent, are not commonly used as a color temperature above 3500K is considered too cool of a light for followspots, according to OCCE. LED
Spotlights are also not typically implemented as high output LEDs are also relatively new. For example, in was not until 2008 that Cree released LED lighting products powerful enough to be used in ceiling fixtures. Groups like OCCE and CTC also mentioned that LED spotlights are not common on the market and the models OCCE tested in the past were not bright enough for a setting such as the CFAC auditorium.

A better reference for understanding followspots is ETC’s Source Four spotlight, which is a common spotlight used in current theaters. Source Fours use around 375-750W to power their incandescent light sources with output ranging from 7000 to 21000 lumens. The light beam angle can range from 5° to 90° which is set by the lens placed in the lens tube of the device. One Source Four model in particular called the Source Four Zoom has tunable lenses with beam angle ranges from 15-30° or from 25-50°. This allows the user to change the beam angle remotely and without swapping lenses. The reflectors on the inside of the device are dichroic, meaning that 90% the infrared output from the light source reflected by the reflector is removed. Lastly, color temperature can be changed to a less desirable but still acceptable 3050K (from 3250K) in order to increase lamp’s life from 300-400 hours to 1500-2000 hours, which is the implementation that Calvin’s OCCE uses to save money.

6.2 Lighting Subsystem Overview

The lighting subsystem has the express purpose of creating the beam of light to be centered on a subject on stage. This involves performing multiple duties. First, it receives commands from the user, such as on or off or dim, passed from the actuation microcontroller and sending status information, such as overheat errors, to the user by passing the information to the actuation microcontroller, as seen in Lighting Subsystem Overview. It executes those commands through a controller. Lastly, it involves the optical and lighting duties to create the light output and shape it correctly. To better understand the design of this subsystem, the light system design can be thought of as four smaller systems: the thermal management system, the optical system, the electrical system, and software system.
The thermal management system includes the LED array and the heat it creates along with the components involved in removing that heat from the overall system. The LED array’s lifespan is reduced based on the junction temperature of the LED array. An increase in junction temperature from 75°C to 150°C can reduce the operation lifetime by more than 70%. Therefore, a heat sink shall be included to remove the heat generated from the LED which is 75% of input power for a Bridgelux Vero Chip-on-Board (CoB) LED array. So, a heat sink is considered in the thermal system as well as a thermal interface material between the LED and heat sink. A thermal interface material (TIM) reduces the amount of air pockets in between the LED and heat sink. Figure 8 shows the thermal system model where thermal resistance can be measured for certain components of the thermal system and junction temperature can be estimated. The junction temperature is the temperature of the junctions of the LED’s in the array. The case temperature is the temperature of the metal plate bottom of the LED array which contacts the TIM. The heat sink temperature is the temperature measured at the surface of the heatsink in contact with the TIM. Lastly, ambient temperature is the temperature of the surrounding air.
The optical system includes the components that create and manipulate the light output. The LED array generates the light output, but it does not do so in a focused manner. To reclaim light output not directed forward by the LED array, a reflector is included which reflects light forward. The reflector also provides the desired focus for the beam. Since the reflector only focus light that is reflects, some light output is still not at the desired beam angle. A stray light cancelling attachment is included that passes light at the correct angle and blocks light at angles larger than the reflector beam angle. The end result will be a created beam of light that is 5-10 ft in radius and between 50 and 100 footcandles on stage as stated in the system requirements chapter of the report. Figure 9 shows the light as it is reflected and directed from the LED light source.

The electrical system includes the devices that power and control the lighting subsystem. The LED requires a specific forward voltage and forward current. Therefore, an LED driver is required in the system to provide the LED required voltage and current as well as to offer a way to adjust the current for dimming purposes. Dimming is achieved through resistive dimming method, so a digital potentiometer is used to allow the microcontroller to control the LED driver output. User controls for light levels as well as states of on or off shall be able to affect the LED driver to change its output. These signals shall be able to be interpreted and executed in the subsystem, so the system requires a microcontroller. The signal shall be sent from the actuation controller to the lighting microcontroller which requires a line of
communication. The microcontroller receives power from a 12V 5A power supply with barrel jack connector and 110VAC outlet plug. The driver receives power from a 110 VAC outlet. A relay is required for the current output to control the light ON/OFF status. The heatsink is powered from the microcontroller power supply with 12V DC regulated down to the necessary 5V DC. Figure 10 shows the electrical system components and how they interact with one another.

![Electrical Component Block Diagram](image)

**Figure 10: Electrical Component Block Diagram**

The software system involves the software programmed onto the microcontroller that accepts and sends signals to the actuation microcontroller and subsystem devices. The software shall interpret those signals and make corrections to the correct output pins. Temperature data read from the temperature sensor shall also be interpreted to make the proper adjustments such as a shut down due to overheat. Lastly, the software shall determine what information to send as status information to the user.
6.3 Lighting Subsystem Hardware

6.3.1 LED Light Source

6.3.1.1 Requirements
The main requirement for the LED light source is light output. As stated in the system requirements, the LED shall have no less than 4000 lumen output in order to provide 50-100 foot candle illuminance at the stage with a beam of at least 5ft radius. Lastly, the color temperature shall be in the range from 3000K to 3500K.

6.3.1.2 Design Criteria for LED Light Source
Design criteria for the LED technology include cost, light output, and implementation with heat sink. The cost considers the cost of the emitter or group of emitters that meets or exceeds the lumen output requirement. The light output is considered lumens greater than or equal to 4000 for single package emitters. Implementation with a heat sink considers how connections to the heat sink are made.

6.3.1.3 Design Alternatives for LED Technology
A common implementation for new LED spotlight models is a design that uses multiple surface mount LEDs. ETC’s LED Source Four model utilizes 60 Luxeon Rebel ES surface mount LEDs to achieve adequate light output.\(^{27}\) One advantage to this implementation is that a designer can simply add more LEDs to the design to get more light output, which makes the design flexible. The ETC example shows that the design allows high outputs to be achieved considering a single Luxeon Rebel can produce 100 lumens per LED. The main disadvantage to this design is that the price of each LED can be between $2-4 each, which means a design that meets the output requirements can quickly exceed $100. So, to meet the lumen output requirement of 4000 lumens, 40 LEDs would be required at a cost between $80-160. Also, since the layout of these LEDs varies, matching components such as reflectors and heat sinks are not common. The board for which the LEDs are mounted would need to be fabricated to match a specific heat sink, then connected using either screws (if threaded screw holes are in heat sink), thermals paste, or some other thermal adhesive.

A more recent design alternative is chip-on-board (CoB) LED arrays. These devices include a large amount of LEDs built into a surface and sold as a single package. Thus, the LED arrays can be much smaller while still creating adequate output. In fact, LED arrays can produce output around 1000-16000 lumens. Since most implementations of these devices require only a single package, compatible reflectors and heat sinks are available for most models. Lastly, costs tend to range around $30-60 which is far below
the $100 or more for the surface mount approach. The disadvantage of the design alternative is the added in-flexibility of a design built around an LED array. If the light output of a high output model is insufficient for the application, a different model will have to be ordered or multiple will have to be put into the design which negates the component compatibility advantage. A heat sink can be attached to a CoB connector with screws or directly to the emitter with a thermal adhesive.

The last design alternative is LED arrays built around high flux density LEDs. This design is most recognizable in Philips’ Luxeon S3000 and S5000 line of LED arrays. The emitters are composed of a cluster of high flux density LEDs mounted to a surface in a singular package. While this design shares the many benefits with CoB LED arrays, the cost for such devices is around $90-100 per package for output around 3000-6000 lumens. The heat sink is also attached through connectors or thermal adhesives like the CoB array.

6.3.1.4 LED Technology Decision
CoB LED array technology is the preferable choice due to significantly lower costs, simpler implementation with heat sinks, and higher light output. CoB LED arrays are the least expensive option outlined, at $30-60 instead of around $100 or greater for high-flux density or surface mount options. Also, CoB options well exceed the output requirements of above 4,000 lumen output with outputs ranging up to 11,000 lumens for the Cree CXA3070 model. Also, parts like reflectors and radial heat sinks are often desired around and tested for CoB arrays but not as often for clusters of surface mount LEDs.

6.3.1.5 CoB LED Design Criteria
The design criteria for CoB LED include cost, color temperature, and light output. Color temperature shall be within the 3000-3500K range. Light output shall exceed 4,000 lumens. Lastly, implementation options such as attachment to heat sinks as well as reflectors shall be considered.

6.3.1.6 CoB LED Design Alternatives
The CoB LED design alternatives are shown in
and show companies with parts available in single quantities on retail sites such as DigiKey or Mouser. Implementation details are considered between top choices due to the time to research component compatibility and are therefore left off the table.
Table 5: LED Module Alternatives

<table>
<thead>
<tr>
<th>Company</th>
<th>Model</th>
<th>Part Number</th>
<th>Color Temperature</th>
<th>Output (lumens)</th>
<th>Price</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cree</td>
<td>XLamp CXA</td>
<td>CXA2540</td>
<td>3000-3500K</td>
<td>4000 to 5000</td>
<td>$30.00</td>
<td>Mouser</td>
</tr>
<tr>
<td>Cree</td>
<td>XLamp CXA</td>
<td>CXA3050</td>
<td>3000-3500K</td>
<td>5000 to 6300</td>
<td>$36.00</td>
<td>Mouser</td>
</tr>
<tr>
<td>Cree</td>
<td>XLamp CXA</td>
<td>CXA3070</td>
<td>3000-3500K</td>
<td>8000 to 1100</td>
<td>$40.00</td>
<td>Mouser</td>
</tr>
<tr>
<td>OSRAM</td>
<td>SOLERIQ E</td>
<td>GW KALRB3.EM</td>
<td>3000K</td>
<td>4500</td>
<td>$70.00</td>
<td>Mouser</td>
</tr>
<tr>
<td>Bridgelux</td>
<td>RS</td>
<td>BXRA-30E4000-H-03</td>
<td>3045K</td>
<td>4925</td>
<td>$35.00</td>
<td>Digikey</td>
</tr>
<tr>
<td>Bridgelux</td>
<td>RS</td>
<td>BXRA-30E7000-J-03</td>
<td>3045K</td>
<td>7850</td>
<td>$55.00</td>
<td>Digikey</td>
</tr>
<tr>
<td>Bridgelux</td>
<td>RS</td>
<td>BXRA-30G4000-H-03</td>
<td>3045K</td>
<td>4200</td>
<td>$37.00</td>
<td>Digikey</td>
</tr>
<tr>
<td>Bridgelux</td>
<td>RS</td>
<td>BXRA-35E4000-H-03</td>
<td>3465K</td>
<td>5300</td>
<td>$35.00</td>
<td>Digikey</td>
</tr>
<tr>
<td>Bridgelux</td>
<td>RS</td>
<td>BXRA-35E7000-J-03</td>
<td>3465K</td>
<td>8400</td>
<td>$55.00</td>
<td>Digikey</td>
</tr>
<tr>
<td>Bridgelux</td>
<td>Vero</td>
<td>Vero 29</td>
<td>3000-3500K</td>
<td>7400 to 9400</td>
<td>$46.00</td>
<td>Digikey</td>
</tr>
<tr>
<td>Sharp</td>
<td>Mega Zenigata</td>
<td>GW6DME35NFC</td>
<td>3500K</td>
<td>4500</td>
<td>$19.00</td>
<td>Mouser</td>
</tr>
<tr>
<td>Sharp</td>
<td>Mega Zenigata</td>
<td>GW6DME30NFC</td>
<td>3000K</td>
<td>4350</td>
<td>$20.00</td>
<td>Mouser</td>
</tr>
</tbody>
</table>

6.3.1.7 CoB LED Design Choice

The array that will be used in the design is the Bridgelux Vero 29 LED array shown in Figure 11, whose high output models range between 8,000 and 11,000 lumens, is well above the 4000 lumen requirement. Extra light output allows for flexibility in designing the beam angle and reflectors, as wider beam angles and less efficient reflectors reduce the light output. The Vero 29 offers the high output of the options while costing around the same price as arrays with around 2000-3000 less lumens, such at the Bridgelux RS or OSRAM models. Lastly, unlike the CXA3070, the Vero is compatible with some heat sinks.
available on Mouser and Digikey, such as the Nuventix models discussed below, while the CXA3070 is not.

![Figure 11: Bridgelux Vero 29 LED Array](image)

6.3.1.8 **CoB LED Implementation**

A connector contains the positive and negative leads for the LED array which are then connected to the constant current outputs from the LED driver. The reflector is screwed to the LED through the same screw holes that connects the LED array to the heat sink. Thermal grease is used as the thermal interface material which removes air pockets between the two components to decrease thermal resistance.

6.3.2 **LED Driver**

6.3.2.1 **Driver Requirements**

The driver shall be able to output a minimum of 2A forward current and at least a 43V forward voltage as specified by the Vero 29 LED array datasheet. The driver shall also be constant-current or constant-current-constant-voltage LED driver. A constant voltage driver can achieve the desired output current and voltages; however, because small changes in forward voltage result in large changes in current and current determines the output of the LED, constant voltage LED drivers can cause unsteady or jittery output. Lastly, the driver shall come in a module package as opposed to an IC design. LED driver IC’s require designing and building a circuit whereas the output, dimming, and circuit protection considerations are included in module implementations. In order to create a prototype reliably and in time, the module implementation is required.
6.3.2.2 Driver Design Criteria

The design criteria for the LED driver include output voltage range, dimming capabilities, output current, and cost. Greater output voltage ranges allow for greater reliability because output voltage for constant current outputs varies, especially with temperature variations. For example, a constant current output of 2.1 A has an output range from 34.7V to 42.5V. But, for case temperature operating limits, the range is 32.4V to 43.5V. For dimming capabilities, PWM allows for microcontroller output to directly work with the LED driver whereas 0-10V or resistive dimming require extra components to control the LED driver output. If the driver does not have dimming capabilities, a separate circuit must be designed to handle dimming. Therefore, a dimming option labeled as “none” would represent the most complex option. Therefore, PWM is preferable while 0-10V and resistive are still workable. No dimming capabilities would be the last option. For output current, more current means more light output from the LED. The max recommended output for the Vero 29 is 2.8A.

6.3.2.3 Driver Design Alternatives

shows the available drivers that fit the design requirements for the LED drivers with options sold from retailers Digikey, Avnet Express, and Future Electronics.

<table>
<thead>
<tr>
<th>Make</th>
<th>Model</th>
<th>Output Current</th>
<th>Output Voltage</th>
<th>Dimming</th>
<th>Price</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aimtec</td>
<td>AMER120-50250AZ</td>
<td>2.5</td>
<td>36-50</td>
<td>pwm, resistive, 0-10Vdc</td>
<td>$63.18</td>
<td>Avnet Express</td>
</tr>
<tr>
<td>Recom</td>
<td>RACD100-48</td>
<td>2.1</td>
<td>34-48</td>
<td>none</td>
<td>$85.40</td>
<td>Digikey</td>
</tr>
<tr>
<td>Inventronics</td>
<td>EUC-096S210DT</td>
<td>2.1</td>
<td>22-54.7</td>
<td>0-10v, resistive</td>
<td>$59.41</td>
<td>Future Electronics</td>
</tr>
<tr>
<td>Inventronics</td>
<td>EUC-096S210DV</td>
<td>2.1</td>
<td>22-45.7</td>
<td>0-10v, resistive</td>
<td>$69.96</td>
<td>Future Electronics</td>
</tr>
</tbody>
</table>

6.3.2.4 Driver Design Choice

The driver chosen for the project was the Aimtec model. It was chosen for the high max forward voltage which ensures that the LED array does not turn off without expressly being controlled to do so. It also has the highest current output of the options at 2.5A as opposed to 2.1A like the other models. For the 3500K
Vero 29 LED array, this amounts to an increase of light output from 8650 lumens at 2.1A to 10088 lumens for 2.5A. Lastly, the Aimtec model is the only driver module to have the option of PWM dimming as well as includes 0-10V and resistive dimming options.

6.3.2.5 Driver Design Implementation

The driver has positive and negative output leads in which the LED array connects in series. In order to smooth out the output current, two 10 mF capacitors are placed in parallel to the LED array. The output needs to be smoothed in order to prevent the light from flickering due to changes in forward current. The output is switched with a TE Connectivity RT424005F-ND DPDT relay, which can operate for a max current of 8A.\(^3\) The LED driver output board is shown in Figure 12. The BOM of the circuit is available at http://www.calvin.edu/academic/engineering/2013-14-team9/links/LightingSubsystemBOM.pdf. This allows the user to turn the LED on or off with the microcontroller setting the voltage input for the switch. The 120VAC input wires are spliced with an outlet plug and cord. This allows the LED driver to be turned on or powered when the cord is plugged into an outlet. Dimming is performed through the resistive dimming method where a resistance is in between the dimming terminals selects the output current. The resistive range that the LED accepts is between 18K and 0K with 0K being the lowest output setting and 18K being the highest output setting. The resistance is determined by the microcontroller setting. The microcontroller sets the 8-bit resistance setting for a digital potentiometer. The digital potentiometer is the Microchip MCP4261-103E Dual SPI Digital POT, which has a resistance range of 0-10K with 255 steps.\(^3\) The dimming board is shown in Figure 13. Since the current output and resistance relationship are logarithmically related, the greatest change in current output takes place at lower resistances. Also, the bottom range resistance does not reach exactly zero. So, the dual outputs are connected in parallel to cut the resistance in half to increase the output current range and thus the actual dimming range. Lastly, to ensure the system does not overheat, a temperature sensor is placed to measure the temperature at the passive cooling heat sink. This temperature is then used to estimate the case temperature. Criteria for the temperature sensor include mounting and temperature measurement range. Preference is given to indirect measurement as mounting a sensor to the heatsink will be more difficult than mounting to the enclosure surface as a heatsink may be unable to be used with a drill press. The temperature sensor chosen was the Melexis MLX90614 IR temperature sensor. The temperature sensor board is shown in Figure 14. Because the sensor measures temperature through infrared, the sensor does not need to be mounted in direct contact with the heat sink. Thermistors and thermocouples were considered as design alternatives; however, they must be mounted to the heatsink. The sensor can measure object temperature in the range of -70C to 380C, which encompasses the 105C case temperature maximum.\(^3\)
Figure 12: LED Driver Output Board

Figure 13: Dimming Control Board
6.3.3 Heat Sink

6.3.3.1 Heat Sink Requirements

Bridgelux estimates the heat loss for a warm white Vero 29 LED array to be 75% of input power. The maximum voltage expected at typical case temperature of 85°C is about 43V. The maximum output current anticipated is 2.575A (+/- 3% error from stated 2.5A). Therefore, when operating the LED driver at maximum output setting the anticipated heat loss is approximately 80W (40V * 2.5A). At typical operation with the LED driver the heat loss is 73W for typical output current of 2.5A at 39V. For the 2.1A standard listed operation current, the maximum heat loss is 67W (for 42.5V). The typical heat loss at 38.6V is 61W. Since the LED driver can reduce output as specified, the thermal wattage of the heat sink shall then be subject to the maximum anticipated heat loss at standard output current of 2.1A. So, the required thermal wattage of the heat sink shall be no less than 67W. Thermal resistance must also be taken into consideration. This determines what the case temperature will be. The maximum case temperature for the Vero 29 LED array is 105°C. The thermal resistance is calculated with the equation (ambient air temperature - temperature of LED array case) / LED heat output. At 2.1A, the heat output is 67W. The typical case temperature, as stated by Bridgelux, is 85°C. The maximum expected ambient air temperature is 38°C (100°F). So, the thermal resistance shall be no more than 0.70 C/W. For LED driver maximum output at 2.575A, the thermal resistance shall be not more than 0.59 C/W. Lastly, the maximum noise produced by the heat sink is 30 dBA as the max is 30 dBA for the system.

6.3.3.2 Heat Sink Design Criteria

Design criteria include cost, thermal wattage, thermal resistance, noise, and Vero 29 attachment compatibility. Thermal wattage determines the amount of heat that the system can remove and must be greater than the amount output by the LED array. The thermal resistance determines what the case temperature will be in relation to the ambient air temperature with lower resistance contributing to a lower case temperature. Noise is the noise output by the system in dBA. Lastly, compatibility is whether or not
the heat sink has screw-hole positions and sizes that match the mounting screw holes of the Vero 29 LED array. It is easier to work with preconfigured holes as opposed to drilling and tapping into one that does not have them.

6.3.3.3 Heat Sink Design Alternatives

A few alternative include Nuventix hybridized designs. The systems include a synthetic jet active cooling device and a passive cooling heat sink to which the LED array is attached to. The larger passive heat sink allows for multiple synthetic jets to be attached to it, allowing for greater thermal wattage if desired. The other option is a Sunon hybridized design which uses a fan for active cooling instead if a synthetic jet. The design alternatives are shown in Table 7.

Table 7: Cooling Design Alternatives

<table>
<thead>
<tr>
<th>Make</th>
<th>Model</th>
<th>Type</th>
<th>Vero 29 Screw Dimensions</th>
<th>Noise (dBA)</th>
<th>Thermal Resistance (C/W)</th>
<th>Thermal Wattage (W)</th>
<th>Price</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nuventix</td>
<td>NX302105 and NX202100</td>
<td>synjet (1) and heatsink</td>
<td>yes</td>
<td>22</td>
<td>0.42</td>
<td>95</td>
<td>$109.78</td>
<td>Digikey</td>
</tr>
<tr>
<td>Nuventix</td>
<td>NX302105 and NX202100</td>
<td>synjet (2) and heatsink</td>
<td>yes</td>
<td>24</td>
<td>0.32</td>
<td>125</td>
<td>$136.2</td>
<td>Digikey</td>
</tr>
<tr>
<td>Nuventix</td>
<td>NX302105 and NX202100</td>
<td>synjet (3) and heatsink</td>
<td>yes</td>
<td>26</td>
<td>0.27</td>
<td>148</td>
<td>$162.62</td>
<td>Digikey</td>
</tr>
<tr>
<td>Nuventix</td>
<td>NX200106 and NX300160</td>
<td>synjet and heatsink</td>
<td>yes</td>
<td>22</td>
<td>0.49</td>
<td>82</td>
<td>$51.12</td>
<td>Digikey</td>
</tr>
<tr>
<td>Sunon</td>
<td>LM310-001A99DN</td>
<td>fan with heatsink</td>
<td>no</td>
<td>17.8</td>
<td>0.33</td>
<td>(Tested for 180W LED)</td>
<td>$41.53</td>
<td>TME</td>
</tr>
</tbody>
</table>
6.3.3.4 Heat Sink Design Choice

The heat sink chosen was the Nuventix NX300160 paired with the NX200106. It is significantly lower in cost to the other Nuventix options which each cost over a hundred dollars. While the heat sink is more expensive and has less cooling power than the Sunon option, the chosen heat sink is compatible with the LED array which reduces the chance of accidentally breaking the heat sink trying to drill holes as well as reducing the time to build the spotlight. Since this model is a prototype, thermal wattage and resistance were secondary to mounting holes as having matching screw holes reduces the build time. In a production model, the price as well as a safe thermal wattage and thermal resistance safety factor would be considered more important as mounting dimensions as dimensions could be worked out with the manufacturer ahead of time.

6.3.3.5 Heat Sink Design Implementation

The heat sink is set at the manufacturer termed “Standard Operation” setting where the power line is set to 5V and GND line is connected to GND but the PWM and heartbeat lines not used. The system is powered through 12V power supply used by the microcontroller. The 5V pin on the microcontroller can only provide 100 mA while the heatsink needs 150 mA. So, a voltage regulator is used to step down the 12V output from the V\textsubscript{in} pin on the microcontroller to 5V. With standard operating setting thermal resistance of 0.54 C/W, maximum ambient air temperature of 38C, and maximum heat generation of 80W, the expected maximum case temperature of the LED array is 84.4C which is well below the maximum operating case temperature of 105C.

6.3.4 Controller

6.3.4.1 Controller Requirements

The controller shall have both I\textsubscript{2}C and SPI pins. I\textsubscript{2}C may be used for the temperature sensor and SPI is used for the digital potentiometer. The microcontroller shall be compatible with the Arduino development environment to keep the controller code used in the actuation and lighting subsystems the same. This is due to a design decisions for the actuation controller and beacon being made before the lighting subsystem design decision. The system shall exist in a development board implementation to eliminate the need to design, fabricate and test an additional circuit. A production model would most likely use a custom board design to eliminate board components that are not needed for the design. Also, the price of the ATmega Microcontroller IC is cheaper than a development board, costing around $3.23 on Mouser as opposed to $25-50 for a full development board such as an Arduino Uno or Sparkfun RedBoard. The microcontroller shall have mini-USB or micro-USB port for programming in order to use existing cables.
Lastly, the operating voltage shall be 5V as opposed to 3.3V. Devices such as regulators and temperature sensors typically require 5V or 3.3V. Arduino microcontrollers that require 5V have both 5V and 3.3V pins while those that require 3.3V only have 3.3V pins as 5V can be regulated down to 3.3V while 3.3V cannot be regulated up to 5V.

6.3.4.2 Controller Design Criteria

The design criteria for choosing the controller include the number of GPIO pins (analog and digital pins), PWM pins, barrel jack power input, and cost. More GPIO pins allow for more flexible development as a change in the necessary connections in the subsystem design can be accommodated. PWM pins are not required for the design; however, some options, such as heat sink control, can be explored with PWM capabilities. Available power supplies for microcontrollers have barrel jacks, so a barrel jack input is desired. If a barrel jack is not included, a separate circuit would need to be created to convert to the necessary connection feature.

6.3.4.3 Controller Design Alternatives

Prominent Arduino design alternatives are displayed in Table with prices gathered from both Amazon and Sparkfun Electronics.

<table>
<thead>
<tr>
<th>Make</th>
<th>Model</th>
<th>Analog In</th>
<th>Digital I/O</th>
<th>PWM</th>
<th>Barrel Jack</th>
<th>Amazon</th>
<th>Sparkfun</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino</td>
<td>Uno</td>
<td>6</td>
<td>14</td>
<td>6</td>
<td>yes</td>
<td>$27.95</td>
<td>$29.95</td>
</tr>
<tr>
<td>Arduino</td>
<td>Leonardo</td>
<td>12</td>
<td>20</td>
<td>7</td>
<td>yes</td>
<td>$20.01</td>
<td>$24.95</td>
</tr>
<tr>
<td>Arduino</td>
<td>Mega 2560</td>
<td>16</td>
<td>54</td>
<td>15</td>
<td>yes</td>
<td>$45.00</td>
<td>$58.95</td>
</tr>
<tr>
<td>Arduino</td>
<td>Micro</td>
<td>12</td>
<td>20</td>
<td>7</td>
<td>no</td>
<td>$25.95</td>
<td></td>
</tr>
<tr>
<td>Arduino</td>
<td>Nano</td>
<td>8</td>
<td>14</td>
<td>6</td>
<td>no</td>
<td>$10.65</td>
<td></td>
</tr>
<tr>
<td>Sparkfun</td>
<td>RedBoard</td>
<td>6</td>
<td>14</td>
<td>6</td>
<td>yes</td>
<td></td>
<td>$24.95</td>
</tr>
</tbody>
</table>

6.3.4.4 Controller Design Choice

The Sparkfun RedBoard, an Arduino Uno clone, was chosen. While the price listed is $24.95 on Sparkfun, the actually cost for the board is nothing as a RedBoard was already available to use. Also, unlike the Micro and Nano options, the RedBoard has a DC barrel jack power input. Lastly, the Redboard is less expensive, in reference to price, than the Uno it is designed after and the Mega 2560. If the
Sparkfun RedBoard had not been available, the Arduino Leonardo would have been chosen as it has a DC barrel jack, has the lowest price of boards with DC barrel jacks, and is only surpassed in pins counts by the Arduino Mega 2560. The Nano has the lowest cost, but the number of pins is lower than the Leonardo.

6.3.4.5 Controller Design Implementation

The microcontroller is powered by a 600mA, 12V power supply with barrel jack connection. Since a USB can only supply 500mA, a power supply which can provide 600mA should be more than sufficient. Communication with the digital potentiometer is handled through SPI, which demands pins 10 through 13. Communication with the temperature sensor is handled through I2C, which demands pins A4 (SDA) and A5 (SCL). The 12V output pin, $V_{in}$, is connected to a 5V regulator with the 5V line powering the synthetic jet in the heatsink.

6.3.5 Spotlight Mounting

6.3.5.1 Mounting Requirements

The mounting method shall support at least 400 lbs. This ensures that if someone puts their weight on the spotlight it will not break or come loose if a person combined with the spotlight weight is estimated to be up to 400lbs. Also, the space that the system uses shall be no more than half of than 2ft of catwalk space to allow people to pass by.

6.3.5.2 Mounting Design Criteria

The mounting design criteria include cost, portability, and space required. The space required is measured as the greatest width of the pathway cross-section that the mounting design requires (i.e. how much walking space width is lost in the path). Since the mounting design will take up walking space, a lower cross-section width of the path is desired. Portability is how the spotlight and its mounting design are moved from place to place.

6.3.5.3 Mounting Design Alternatives

The overall assembly will need to be held in place for a standard catwalk setting. A few methods that have been considered are stands, c-clamps, and hooks. A stand is set up on a flat surface with the spotlight held up at the top. The c-clamp holds the spotlight from a batten from which the spotlight hangs downward. Some benefits of a stand include easier access to the user and mobility on a catwalk. However, catwalks have confined space and a stand may be inconvenient for staff using the catwalk. A c-
clamp has the added benefit of hanging on the outside of the catwalk which takes up none of the catwalk space. Also, adding a c-clamp to the design is far simpler than building or ordering a suitable stand. Hooks attached the spotlight to a batten but do not tighten around the batten as c-clamps do.

6.3.5.4 Mounting Design Choice

In terms of portability, clamps require the most work to move. This is because the clamps must each be unscrewed and reset in the new location. The hook, stand, and tripod design only need to be picked up and set down in the new position. Clamps and hooks also have less portability than tripods and stands as they require a compatible railing to hook or latch onto while stands and tripods only require a flat surface. In terms of space required, the clamp and hooks models require little to no space in the walkway because the spotlight hangs on the outside of the walkway. Lastly, in terms of cost, the stand is the lowest. This is because the stand can be built using wood or metal materials in the shop was well as the screws required. Clamp and hook designs require purchasing the needed clamps and hooks to complete the design. Therefore, due the ability of the stand to be fabricated in the shop without the need to purchase extra materials and its added portability over the hook and clamp designs, the stand design was chosen as shown in Figure 15. In a production design, materials will need to be purchased. Since the hook design is more portable than the clamp design and use less room than the stand design, a hook design would be the production design choice.

Figure 15: Spotlight with Stand
6.3.5.5 Mounting Design Implementation

The spotlight enclosure is attached to a horizontal bar, as shown in Figure 15. The horizontal bar is attached between the two legs of the stand with one screw set in the center of the bar to allow the spotlight set in the desired angle and a second screw to hold the position the user desires. Each leg of the stand is has a single foot with cross-bracing between the legs to keep the stand stable when the spotlight moves or a user bumps the design. The power cords of the spotlight are plugged into a surge protector attached to a leg of the stand with an extension cord running to the nearest outlet. The surge protector means only one outlet plug is required to power the overall system.

6.3.6 Parabolic Reflector

6.3.6.1 Reflector Requirements

A parabolic reflector is shown in Figure 16. The opening diameter for the LED shall be no less than 29.2 mm so that no part of the reflector overlaps with the LED array. The beam shall have a radius of 5 ft with center referenced from the waist of the subject. This gives a radial margin of error of approximately 2 ft for a 6 ft individual on stage. This is to be measured at the closest position on stage to the followspot which happens to be approximately 50 ft for a venue such as Gezon Auditorium. The beam diameter in feet can be determined using the formula \( \text{diameter} = 2 \times \tan(\text{beam angle}) \times \text{distance} + \text{reflector output diameter} \). The reflector output diameter is the diameter of the reflector at the opening edge and is 0.38 ft. With a distance of 50 ft and diameter minimum of 10 ft, the beam angle needs to be
greater than 5.5 degrees in order to make sure that at any point on stage the target is sufficiently covered. With maximum light output at approximately 10088 lumens, the maximum beam angle that can be used while maintaining 50 footcandles for illuminance is about 9 degrees.

6.3.6.2 Reflector Design Criteria
The design criteria for the reflector include beam angle, LED mounting compatibility, and cost. A smaller beam angle will mean greater illuminance at same distances from the spotlight because illuminance is measured in lumens/ft². LED mounting capability is whether or not the reflector has screw holes in corresponding locations to the LED array mounting screws.

6.3.6.3 Reflector Design Choice
Only one parabolic reflector was found that meet the LED array diameter and beam angle range. The reflector chosen was the LEDiL ANGELA S. The reflector has a beam angle of 7.5 degrees. The opening diameter is 32 mm which is greater than the required 29.2 mm. The screw-hole locations do not line up with the LED array mounting holes. The cost of the reflector is $10.70.

6.3.6.4 Reflector Design Implementation
Because the mounting screw holes do not line up, holes were drilled into the reflector to match the location and screw size (M3) of the LED mounting holes. Then, the reflector could be mounted directly to the LED array, ensuring that the focal point of the reflector and the LED array matched. While the reflector creates the correct beam angle for reflected light, light that is not reflected can still be at beam angles larger than the reflector’s, which causes the beam to appear soft or not well defined. Therefore, an attachment was created to block the stray light while passing the reflected light as shown in Figure 17. The focal point of the attachment is in line with the reflector focal point, so that the reflected light is passed through. Next, successive walls at angles corresponding to the beam angles for the particular beam diameter serve to block light coming directly from the LED that is at angles larger than the beam angle. The bottom face of the attachment screws into the base plate of the spotlight. The 3D-printed attachment cost approximately $110 to fabricate using the 3D printer in the engineering building at Calvin College.
6.4 **Lighting Subsystem Software**

System requirements should be considered when making design decisions in regards to the lighting system control software. First, the software language and development environment shall be compatible with the Sparkfun RedBoard microcontroller. Arduino’s programming language was chosen to program the development board along with Arduino’s IDE. While the IDE can program multiple languages onto the microcontroller, Arduino’s language was chosen due to its similarity to C++, with which everyone in the group is familiar, and to preserve similarity between subsystems. Existing code amongst the microcontrollers used in the actuation subsystem is written in Arduino’s programming language. The system shall also be reliable to meet the design norm of trust. Therefore, the software shall turn off the light if a case temperature over 100°C is reached, as the LED array stops functioning if a case temperature of 105°C is reached. Anything above the maximum current could damage the LED or reduce its lifespan.

<table>
<thead>
<tr>
<th>Pins Defined in Software</th>
<th>Pin Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON/OFF (onoff)</td>
<td>8</td>
</tr>
<tr>
<td>Actuation IN (actIn)</td>
<td>7</td>
</tr>
<tr>
<td>Actuation Out (actOut)</td>
<td>4</td>
</tr>
</tbody>
</table>
The software design includes some inputs and outputs with defined pins shown in Table 9. The inputs for the software include dim setting, on/off, and temperature sensor readings. The dim setting is the desired light output for the spotlight, sent from the GUI to the actuation controller, then passed to the lighting controller through TTL Serial connection (protocol outlined in communication section). The dim command is a received “I” character with the following byte representing the desired 8-bit proposed dim setting. The ON/OFF input is the command from the user telling the lighting system to turn the light on or off. An “F” character message is the command that tells the controller to turn the light off. An “O” character is the command to that tells the controller to turn the light on. The temperature sensor reading is the temperature measured at the heat sink surface which is sent to the microcontroller over I2C with information gathered through analog pins 4 and 5. The heat sink surface temperature can then be used to estimate the case temperature. The outputs of the system are the on/off signal and potentiometer resistance (for dimming). The on/off pin, pin 8, is either set HIGH or LOW which flips the switch in series of the LED array input power line. The resistance output is the output sent to the digital potentiometer to control the resistance seen by the LED driver between the dimming lines. The resistance signal is an 8-bit value corresponding to a linear range of resistance of 0-5kΩ.

A few libraries are also required for the software design. An I2C library provides means for sending and receiving information between the temperature sensor and the microcontroller through an I2C connection. One library in particular, I2CMaster, is compatible with the Melexis MLX90614 temperature sensor. The SPI library is available in Arduino’s library folder. For the digital potentiometer, the McpDigitalPot library is used to control the digital potentiometer with the microcontroller.

Lastly, the lighting subsystem software loop runs continuously on the lighting subsystem microcontroller as shown in Figure 14. The code itself is available on the team website at
http://www.calvin.edu/academic/engineering/2013-14-team9/Code/Lighting_Control.ino. The first component of the software loop is the message check. The software checks if a message has been sent over TTL Serial on Pin 7 and sets respective the respective light status or dim setting variables. In the message reading block of the code, the longest time delay is in regards to intensity readings. If an intensity character is read, the code will keep checking for an intensity byte for a maximum of 100ms. The temperature sensor block involves reading the temperature of the heat sink from the temperature sensor and determining if a shutdown is necessary. If the estimated case temperature is over 100C, then the light status variable is set to off. The maximum settling time for the temperature sensor is 100ms. The temperature check subprogram is derived from open-source code on “bildr.blog,” created to set up a temperature reading test. The light status is changed to off if a temperature greater than 57.0 degrees Celsius is read (calculations to get this value shown in testing section). The ON/OFF block checks to see if the ON/OFF status has been changed and then adjusts the output to the light relay to turn the LED on or off. The DIM block checks the dim setting variable for a change, and then adjusts the 8-bit value output setting to the digital potentiometer to get the desired light level. With the intensity loop maximum time of 100ms and temperature sensor settling time maximum of 100ms, the estimated maximum loop duration is around 200ms.

6.5 Testing and Debug

6.5.1 Subsystem Testing

System testing is to be conducted on various components at different levels of implementation to ensure proper implementation and project performance requirement compliance. The goal is for the determined specifications to be measured and listed, such as illuminance, light output, and power usage, and compared to the earlier system requirements outlined previously. Also, by dividing the testing into small portions then moving to larger portions, it is ensured that smaller components work before implementing them together into larger ones, ending with overall subsystem testing to conclude the testing process.

The first stage of testing was temperature sensor testing. The sensor established communication with the RedBoard microcontroller to verify signals were being sent from the sensor and received from the microcontroller using a form of a “Hello World!” program. The communication can be seen through the IDE serial monitor and is considered successful if temperature readings are shown. The serial monitor successfully displayed room temperature readings of 74.6-74.9F, with care to have nothing in the 90 degree field of view. This is because the temperature read by the temperature sensor is an average of all objects seen in its field of view, so if another object is in the field of view, it will display a temperature
that is not entirely of the desired object. An example of the readings is shown in Figure 19 where the readings show a slightly higher temperature read off of the surface of an arm. To determine if a multiplicative constant needed to be added into the decoding process of the code, the temperature readings was compared to a thermocouple reading with a digital multimeter (DMM) to ensure the readings were within +/- 0.5 degrees of the DMM readings, which is the degree of error of the MLX90614.45 For the 74.6-74.9F readings from the sensor, the corresponding reading from the DMM was 74.5-74.6F which suggests shows that the sensor does not require a multiplicative constant or offset to calibrate readings. Since the test confirms that no offset is seen from the readings, the code is shown to operate correctly in regards to reading data. In production, measuring for offset is not necessary for each device after the code has been varied previously.

Next, the LED driver module was tested, shifting to heat sink testing before turning on the LED array, then continuing with LED driver testing. First, the power input lines were spliced into the power cable to plug into an outlet. Each connection was covered in heat-shrink and the overall splice covered with layers of electrical tape to ensure no shorts occur and a user does not contact exposed wire. Without a load in the output, the voltage across the output was measured as 50.6V. This is the expected output voltage for the constant voltage setting at zero output current. Next, the LED array was connected into the output. Testing with the LED turned on was conducted after testing the heat sink as described in the next paragraph. The LED array turned on as expected without any flickering or brightness issues. The LED driver was then tested for dimming. Resistive dimming was tested first with an analog potentiometer. The LED array dimmed but jittering was observed. Next, the digital potentiometer was used for dimming with
the microcontroller cycling through different step settings to test the dimming range. Step settings lower than 50 caused flickering to be observed from the LED array light output. To fix this issue, two 1000 microF capacitors were connected in parallel in the LED driver output to smooth the current output. This value was determined after testing multiple 50V capacitors in parallel with the output. The capacitance was raised until no flickering was observable at a 0 step value. Also, when reaching step values below 4, the resistance measured stayed the same at approximately 75 ohms. Since the potentiometer has dual outputs, the output were tied in parallel to halve the resistance seen by the LED driver. Since the current to resistance curve is logarithmic, more dimming change occurs at the lower resistance levels, so, by adding more range at the bottom end, an even greater range of light output was achieved in comparison.

The heat sink was the next to be tested. The testing began by driving the device at full power at 5V from a lab power supply. When doing so, the device was shown to turn on and run with the setting at full power. The full power setting is where the power line is supplied by a 5V power line and the ground line is tied to ground. Next, the device was powered with a regulator regulating the 12V output of the microcontroller power supply to 5V. The device turned on and ran at full power as well. Then, the LED was turned on and run until the case temperature reached steady state. The case temperature was measured using the temperature measurement point on the LED array with an offset of +5C to determine the true case temperature. The steady state point was determined when the temperature stopped rising between 5 minute measurement intervals. The steady state time was observed to be approximately one hour. The steady state true case temperature – “true” meaning at the point contacting the heat sink -- was determined to be 81C, well below the 100C threshold. The temperature was read using a thermocouple included in a DMM. The case temperature was measured at a measuring point adjacent to the LED array on the chip with 5 degrees added to estimate the true case temperature. The surface temperature of the heatsink was measured to be 38 degrees Celsius. Therefore, the thermal resistance between the case and heatsink surface is estimated to be 0.54 C/W as determined by the equation \[ \text{thermal resistance} = \frac{81-38}{80W}, \] Using the thermal resistance, the case temperature can be estimated from a heatsink surface temperature reading. For a maximum allowable case temperature, the maximum heatsink surface temperature that can be permitted is 62 degrees Celsius. Therefore, to prevent the system from reaching 105C, a shutoff temperature, as read by the temperature sensor, was set as 57 degrees Celsius. Ultimately, the testing shows that at typical settings the heat sink is more than capable of cooling the LED array.

The next stage of testing involves the master-slave communication. The testing starts by simply confirming that information is being received by the microcontroller over the TTL Serial line. This is confirmed by displaying gathered information from a read method onto the serial monitor. When running
this first test, the characters “F,” “O,” and “I” were observed. This confirmed that not only was data being sent to the microcontroller, the correct set of characters were being sent and read. Next, the dim setting byte value was tested with the message check loop printing a received intensity byte to the serial monitor. When the code was run, values within the 0-255 range were displayed which showed that not only could the actuation controller send intensity values, but it could also send values within the correct range of 0-255. Lastly, the message check loop was integrated with commands to change the state of devices in the lighting subsystem. Light on and off commands and intensity change commands were sent by the actuation controller. Each command was confirmed as the light turned on and off as instructed and intensity change as instructed.

Next, the illuminance was tested. With the spotlight setup in the engineering building on the mezzanine, illuminance readings were taken at varying distances from the spotlight as shown in Table 10. The light readings were taken using an EasyView 33 light-meter. The tilt of the light sensor in relation to the spotlight was adjusted until the peak reading was observed. This means that the sensor is directly facing the spotlight. The desired illuminance goal was approximately 50 fc for a distance about 25-50ft from the spotlight. The actual readings show an illuminance of about 38 fc at 25 ft. Using the calculated beam angles, the light output of the spotlight can be estimated. The average spotlight output was determined to be 1150 lumens. If the spotlight is estimated to output 8000 lumens, 85% of the light output from LED is lost in the spotlight. Possible areas of loss include the reflector, attachment, and mirror. A lower illuminance is also due to the beam angle, 7.5 degrees, being larger than the minimum of 5.5 degrees.

<table>
<thead>
<tr>
<th>Distance (ft)</th>
<th>Illuminance (footcandles)</th>
<th>Estimated Area (ft²)</th>
<th>Estimated Output (Lumens)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>45</td>
<td>25.1</td>
<td>1129.5</td>
</tr>
<tr>
<td>25</td>
<td>30</td>
<td>38.1</td>
<td>1143</td>
</tr>
<tr>
<td>30</td>
<td>22</td>
<td>53.9</td>
<td>1185.8</td>
</tr>
</tbody>
</table>

The last test to be conducted was to test power use. As stated in the system requirements, the spotlight was to use less than 1kW of power in order to demonstrate a commitment to stewardship. When plugging the spotlight into a current meter, the spotlight drew approximately 110W with all devices turned on. This includes the LED, motor controllers, motors, driver, lighting controller, and heatsink. With the emergency stop engaging the system in standby mode, the power drawn was 1.2W. This is likely the heatsink as the
heatsink does not turn off with the emergency switch, since it is needed to cool down the system in a case of overheating.

6.5.2 Debug

For the dimming process, the 8-bit digital potentiometer steps were tested over the full range of values at smaller intervals for lower step settings to correspond with logarithmic relationship with light output. Data intervals of one were used at the lowest steps to determine whether a resistance minimum was hit before 0 step value. The test showed that the 0 step value was the minimum resistance setting, which was determined to be 45 ohms. The data collected shows that all steps observed from 0-255 were accepted and show a linear relationship between step and potentiometer resistance as seen in Figure 20. The graph below also allows current outputs of the LED driver to be estimated as a graph for current output is determined by the resistance seen in the dim line. The current determined from the LED driver graph can then be used to estimate the light output in lumens of the LED array as the light output is a function of forward current as displayed in a graph in the LED array datasheet. It is interesting to note that this graph does show a small step error near the top of the range. This is likely due to a small mismatch between the most significant bit (MSB) of the driver and the sum of the other bits. In this case, the MSB is slightly too small to produce a perfect linear relationship.

![Potentiometer Graph]

**Figure 20: Resistance vs. Step Setting**
To ensure the proper function of the software in the system, software was tested in multiple situations. First, an array of commands were sent from the actuation controller. Beyond the set commands of “O,” “F,” and “I,” invalid commands such as blank commands, disconnected communication lines, and invalid characters were read by the message check block. The invalid commands produced no change in status of the light. Since statuses are confirmed in a case statement and changes to status are made only if a valid commands is seen, the invalid cases fell through the case statement. Since the value of the intensity setting is only in the form of a byte and the intensity can only be set if an “I” command has been received, the dim setting was not altered to an invalid dim setting as seen on the serial monitor.
7 Sensing Subsystem

The role of the sensing system is to detect the actors on the stage and stream this information to the tracking system so that the actors can be located. It consists of a camera and one or more near-IR beacons.

7.1 Camera System

The role of the camera system is to sense the actors and the beacons on stage and provide a view of the stage for the user interface. This system is made up of a camera and a development board.

7.1.1 Development Board

There are many types of Computer Development (Dev.) Board that were feasible for this project. The price range was approximately from $20 to $500 depending on performance and the complexity of the computer board. This board will be used to create a live video stream from the development board with a webcam attached. This stream will be encoded in the board and streamed to other devices such as desktop PC, smartphones, tablets, or laptops. The development boards below were evaluated on their potential frame rate and resolution for streaming video. Higher resolutions and frame rates will lead to better performance from tracking software since the software depends on the accuracy and resolution of the sensing system. Table 11 lists the possible choices of dev. boards with different specifications, and price.

Table 11: Sensing Subsystem Dev. Board Options

<p>| Board          | Hackberry A10 [37] | Raspberry Pi Model B (Revision 2.0) [85] | BeagleBone Black | pcDuino V2 [35] |
|----------------|--------------------|------------------------------------------|------------------|----------------|-----------------|
| Processor (CPU)| 1.0 GHz ARM Cortex A8 | ARMv6 700 MHz (can be overclocked to 1GHz) | Sitara AM3359AZCZ100, 1GHz, 2000 MIPS | ARM Cortex A8 1GHz |
| Memory         | 1GB DDR3          | 512MB DDR3                               | 512MB DDR3       | 1GB DDR3       |
| Boot           | SD card and internal storage | SD card                                 | SD card          | Micro-SD card  |</p>
<table>
<thead>
<tr>
<th>Power</th>
<th>5VDC; DC Jack</th>
<th>5VDC micro-USB plug</th>
<th>Mini-USB or DC Jack; 5VDC</th>
<th>5VDC; USB plug</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>110 x 80 mm</td>
<td>85.6 x 56 mm</td>
<td>86.4 x 53.3mm</td>
<td>125mm x 52mm</td>
</tr>
<tr>
<td>Ethernet Support</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Built-in Wi-Fi</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Software OS</td>
<td>Android 4.0 ICS, Linux support</td>
<td>Raspian Debian Linux</td>
<td>Angstrom Linux, Android, Ubuntu</td>
<td>Linux 3.0, Ubuntu 12.04, Android ICS 4.0</td>
</tr>
<tr>
<td>Video Output</td>
<td>HDMI up to 1080p</td>
<td>HDMI 720p</td>
<td>HDMI 720p</td>
<td>HDMI 720p</td>
</tr>
<tr>
<td>Power Consumption</td>
<td>3.5W</td>
<td>3.5W</td>
<td>2.3W</td>
<td>10W</td>
</tr>
<tr>
<td>Availability</td>
<td>In stock (from Miniand.com)</td>
<td>11,161 in stock for the same day shipping (from Element14.com)</td>
<td>0 in stock for the same day shipping (Digikey.com)</td>
<td>171 in stock for the same day shipping (Sparkfun.com)</td>
</tr>
<tr>
<td>I/O</td>
<td>Simple I/O, UART, USB</td>
<td>GPIO, UART, USB</td>
<td>Simple I/O, UART, USB</td>
<td>GPIO, UART, USB, Arduino extension</td>
</tr>
<tr>
<td>Cost</td>
<td>$65</td>
<td>$35</td>
<td>$45</td>
<td>$59.95</td>
</tr>
</tbody>
</table>
From the table above, the challenge was to select the board that is the most suitable and has the best price to fit within the limited budget. Kevin, having completed research on projects that used these boards to create a video stream, video encoding, and process images, picked the Raspberry Pi (Model B) as the first choice for this project since it is one of the most widely used computer Dev. boards. Moreover, the Raspberry Pi has a large community of users and developers supporting it. Also, it has the lowest price of all the four boards mentioned above, and it is tied for the second lowest power consumption. The team was able to use the Raspberry Pi model B to stream video back to the tracking software well.

7.1.2 Types of Camera

Based on the initial design, the system would have had two cameras: a visual camera and a near-infrared (near IR) camera. A visual camera with the standard IR filter would help the system detect the subject in fully lit circumstances. A near-infrared camera will detect the subject in complete black-out situations by detecting the IR beacon worn by stage performers.

One of the challenges the team faced was times when the beacon was not in the line of sight of the IR camera as could happen when a performer turns around to move, the IR camera would not be able to detect the IR beacon on the actor. To address this issue in part, the team plans to place a beacon on the front and back of each performer. As such it is important for this beacon to be as small and affordable as possible. Particular functionalities and designs of the near IR beacon will be described in section 7 of this report.

Another challenge the team faces is the price of proper near IR camera. The price of a good near-IR camera with high resolution and high frame rate ranges from $1000 to more than $20000. This goes beyond the team’s limited development budget. Therefore, the team’s temporarily solution is to remove the IR filter from an available visual camera, so that the visual camera can also pick up near IR. Table 12 lists possible USB visual cameras the team considered for the development process.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolution</td>
<td>Video capture: up to 1920 x 1080</td>
<td>Video capture: up to 1280 x 720 pixels</td>
<td>Video capture: up to 1280 x 720 pixels</td>
</tr>
<tr>
<td>Sensor</td>
<td>CMOS Sensor Technology</td>
<td>CMOS Sensor Technology</td>
<td>CMOS Sensor Technology</td>
</tr>
</tbody>
</table>

Table 12: Sensing Camera Options
<table>
<thead>
<tr>
<th>Frame Rate</th>
<th>Up to 30 frames per second</th>
<th>Up to 30 frames per second</th>
<th>Up to 30 frames per second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field of View</td>
<td>75°</td>
<td>60°</td>
<td>75°</td>
</tr>
<tr>
<td>Auto-focus</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Cost</td>
<td>$49.99 (from Amazon.com)</td>
<td>$29.39 (from Amazon.com)</td>
<td>$47.82 (from Amazon.com)</td>
</tr>
</tbody>
</table>

Based on the information given in 1, the Microsoft LifeCam Studio had many advantageous features over the Logitech HD C310 such as a higher resolution image and better video quality.

Another choice for visual cameras is an Ethernet/IP camera. These are primarily used as surveillance cameras. They are controlled over the internet, and they can be accessed from many types of device such as PCs, laptops, smartphones or Tablets. Typical IP camera system have some advantages. They are easy to access and use directly over a network, and could be installed by simply plugging them into an Ethernet port instead of going through a single board computer to reach the Ethernet network. They also typically have very good video compression built into the camera. These same compression standards could be hard to replicate on a single board computer without special hardware.

Despite many benefits mentioned above, the cost is the main drawback of using a system of network cameras. A typical system would cost from $875 up to $1000. Also, the prototype system does not yet require such a complex system.

The team’s early testing showed that the tracking software throughput could not handle high resolution images, so the team chose to use the Logitech C310 camera as it was more economical and would provide the same level of functionality. Additional tests showed that the near-infrared beacon could handle more ambient lighting than anticipated, so the team was able to eliminate the need for a visual light camera and visual tracking.
7.1.3 Design Decision

7.1.3.1 Development Board

The major decision that the team made in regards to the sensing system was choosing the Raspberry Pi, shown in Figure 21 over all the other alternatives such as Beaglebone black or PC Duino V2. The reasons why the team made this decision are as follow.

- The Raspberry Pi has an adequate hardware configuration (ARMv6 600MHz processor with 16GB SD card inserted as well as USB 2.0 supported) that was needed for streaming high frame rate video.

- The Raspberry Pi and the Raspbian OS both have considerable support from developers’ communities. Also, troubleshooting data for Raspberry Pi can be found in many forums such as Raspbian, StackOverFlow, or XDA Developer.

- The Raspberry Pi was easily available for testing in Calvin’s computer lab, as new Raspberry Pi boards have been recently purchased by the Engineering Department. Therefore, the team had the opportunity to borrow and use one of these for prototyping.

![Figure 21: Raspberry Pi Model B, with 16GB SD Card Inserted](image-url)
7.1.3.2 Camera

The other decision that the team made for the sensing system was the selection of the final sensor prototype. The team selected the Logitech HD C310 webcam, as team member Phil Fynan had one that could be used immediately. Since the webcam had an IR (Infrared) filter on the lens, it could not detect any significant IR signal. To solve this issue, the team removed the IR filter on the camera. The IR filter is a small piece of colored class over the sensor. The team cracked it during the removal process which led to some spots on the image, but the increased IR sensitivity was well worth the tradeoff. Another issue was that with the normal stream of the camera, the amount of visible light was too intense such that the camera had trouble with detecting the IR source from the tracking beacon (described in section 7.2). The team looked for affordable visible light filters including those made for IR photography. On a forum the team found that the magnetic media from the unused floppy disk could be used as a visible light filter. The team found an old disk and removed a portion of the media. The team taped this filter over the front lens of the camera. This results in the dimming the overall brightness of the video stream while not overly hindering the IR performance. The major drawback of this filter is the degradation of the video fed to the user. The stream is displayed in black and white with dark gamma and blurry video. Figure 22 shows an image of the modified Logitech webcam.

![The Modified Logitech Webcam](image)

Figure 22: The Modified Logitech Webcam
7.1.4 Video Streaming Prototype

The team considered numerous video streaming methods such as FFMPEG/Avconv, GStreamer, MJPG-Streamer, or VLC. Based on the experience of previous project with Professor Kim, Kevin, who is in charge of the camera system, had decided to test the video streaming with two methods: FFMPEG/Avconv and GStreamer.

Avconv is a very fast video and audio converter that can grab from a live audio/video source. It has the ability to convert between arbitrary sample rates and resize video with high quality poly-phase filter. Based on the experience of previous project with Professor Kim, Kevin, who is in charge of the camera system, had decided to test the video streaming with two methods: FFMPEG/Avconv and GStreamer.

The team tested the streaming pipeline in Raspberry Pi with Avconv software. The result is shown in Figure 23 below. The team tested this with RTP streaming by executing the following command in the Raspberry Pi terminal. RTP (Real-time Transport Protocol) is a standardized packet format for delivering audio and video over IP networks. Since the team did not need to play the stream on multiple devices, RTP stream was the optimal choice for this test.

```
  avconv -f video4linux2 -i /dev/video0 -vcodec mpeg2video -r 25 -pix_fmt yuv420p -me_method epzs -b 2600k -bt 256k -f rtp rtp://153.106.112.178:8080
```

To run the pipeline, the following command was executed in the terminal: “avplay rtp://153.106.112.178:8080”. The number “153.106.112.178” corresponds to the Raspberry Pi IP address. This number can be changed depends on the location of the Ethernet port. The number “8080” corresponds to the chosen Port number.

![Figure 23: Video Pipeline Test with Raspberry Pi using Avconv](image)

72
Based on the numbers shown in the terminal above. The resulting average frame rate was 8 fps (frames per second). This result was not adequate to test with the Tracking Software (developed by Phil Fynan). Therefore RTP streaming with Avconv was not a good choice for this project.

The second video streaming protocol investigated was GStreamer. The same method was used for GStreamer software. The team used RTP streaming with JPEG encoded format. The following command was executed to run the pipeline.

```
gst-launch-0.10 -v -e v4l2src ! 'image/jpeg, width=640, height=480, framerate=15/1' ! jpegparse ! rtpjpegpay ! udpsink host=153.106.112.178 port=4000
```

The result of the execution is shown in Figure 24 below.

![Figure 24: Video Pipeline Test with Raspberry Pi using Gstreamer](image)

GStreamer provided a stream with an average frame rate of more than 60 fps, which is beyond the expected and required result. The tested resolution was VGA resolution (640 by 480 pixels). The team tested the delay of the video by waving in front of the webcam. The latency response of the stream was estimated to be less than 100ms, which is acceptable according to the team latency budget. The resulting
stream is shown in Figure 25 below. To play the stream, the team used the User Interface to execute the following command (see section 9 for more details).

```
gst-launch-0.10 -e udpsrc port=5000 caps="application/x-rtp, " 
"media=video, clock-rate=90000, " 
"encoding-name=JPEG, payload=96" " 
"! rtpjpegdepay ! jpegdec ! ffmpegcolorspace ! autovideosink"
```

![Figure 25: Video Streaming Display from the Raspberry Pi using Gstreamer](image)

From Figure X above, the green number is the instantaneous frame rate of the video stream. Given these good test results the team decided that GStreamer was the optimal video streaming method for the Tracking Spotlight system.

Thus, the final camera and streaming portion of the prototype sensing subsystem consists of one Raspberry Pi computer board, one Logitech HD C310 webcam, and a software package using GStreamer to send the video to the tracking subsystem.

### 7.2 Tracking Beacon

The purpose of the near infrared beacon is to give a specially modified camera a target to track when there is little available light. The beacon will flash at a predetermined rate so that multiple beacons can be differentiated and so that beacons can be differentiated from noise which tends to be steady.
7.2.1 Research

There is a good body of previous work that uses IR beacons. The team found a project that uses the IR Beacons and IR Seeker done by Thomas Eng. The IR Beacon he used is made by HiTechnic and is shown in Figure 26. It uses a 9V battery to run three, IR LEDs that flash at a frequency of 1200 Hz. This light could be detected by devices such as the HiTechnic IR Seeker sensor, or even a common digital camera that does not have an IR filter installed. This beacon is shown in Figure 26. Its use gives the team confidence in the capabilities of a near IR beacon and the motivation to build their own beacon.

![Figure 26: HiTechnic Near-IR Beacon](image)

7.2.2 IR Beacon in Spotlight tracking

In this project, the team made an IR beacon similar in function to the one above but with a few modifications. A webcam with the IR filter removed and a visible light filter added detects the IR signal from the beacon. Unlike the HiTechnic beacon, the team’s beacon must flash at a rate lower than the nyquist frequency of the camera for flashing to be detected. If the camera transmits data at 15 frames per second the beacon would need to flash slower than 7.5 Hz for the flashing to be discernable. Because the team did not know the final frame-rate they decided to design the beacon with a user-configurable flashing rate.

7.2.3 Initial IR Beacon Design

The beacon needs to be as small as possible so that it is not awkward for the stage actor to wear. To meet this goal the beacon was kept as simple as possible. The core of the beacon is a moderately high power near-infrared LED. The team investigated the relative performance of 850nm and 950nm LEDs. In testing, both were easily able to saturate the sensor of the modified webcam and were indistinguishable from that video. A production beacon would likely use 950nm LEDs because they are not visible at all to
the human eye while 850nm LEDs are visible as a faint red glow. The team’s prototypes use 850nm LEDs as the slight glow is not very distracting and makes troubleshooting the beacon far simpler. The prototype’s LEDs have a 120° viewing angle. This helps the camera pick them up from a wide variety of angles. The tests conducted by the team showed that this is sufficient to work in most circumstances.

The selected LEDs run at 100mA, and thus for maximum efficiency and battery life the LEDs need a constant current driver. The Texas Instruments TLC5916 8-bit constant current LED sink is used as the LED driver. This chip was graciously provided as a free sample by Texas Instruments in both DIP and SSOP packages.

The final component of the beacon is a small battery. To power the beacon a voltage of 3.3V or more is needed. This is dictated by the requirements of the current sink IC. The overall current draw is determined almost entirely by the LED. The average current draw was measured to be 80mA when flashing at a 50% duty cycle.

The team looked for the smallest available battery that would meet these needs. The team looked into small Lithium Polymer batteries and also small coin cell batteries. Both of these options provide around 3.3V. However the team found that conventional coin cell batteries could not support the 120mA peak current. And rechargeable coin cells that could support this current would not provide adequate runtime. The team thus opted for a very small 400mAH, 3.7V Lithium Polymer battery to power the beacon.

With an average draw of 80mA and a 400mAH battery the team expected to obtain runtimes in excess of four hours. In reality the measured runtime was just over two hours. This early failure was caused by a drop in the battery voltage as the relatively high 120mA current was applied. This high current requirement was one of the leading factors in the selection of a Lithium Polymer battery. The particular battery selected should be able to be support currents up to 800mA according to its vendor. After further investigation the team found the battery in question has a built in short circuit protection circuit. This is an excellent safety feature for a potentially dangerous Lithium Polymer battery, but it could also be contributing an unintended voltage drop. The team believes that a similar battery may exhibit less sag under load and give the full desired runtime of four hours.

The initial beacon prototype used the DIP packaged LED driver and connected via jumper wires to an Arduino Uno. The LEDs had to be surface mounted, so they were mounted on a small piece of copper clad board with the needed traces cut out with a rotary tool. Both an 850nm and 950nm LED were mounted so that their relative effectiveness could be assessed. This prototype can be seen in Figure 27. It
was critical in obtaining early test footage for use in tracking algorithm selection and development, but the team did not feel it was polished enough for their final product and thus set out to make a production quality beacon.

Figure 27: Initial Beacon Prototype

7.2.4 Final Beacon Design

The team’s goals for the final beacon was to replicate the functionality of the first prototype, add a low battery warning, and shrink the beacon so it was no larger than its battery or about 3cm by 4.5cm. To meet these demands the team used almost entirely surface mount parts.

The ATMEGA328 of the Arduino was replaced with an ATTINY85, an 8 pin part. This was challenging given that the 5 IO pins had to be shared between driver control, programming, and battery management. One of these IO pins was used exclusively to measure the battery voltage. The battery voltage is run through a passive voltage divider and then read by the internal analog to digital converter (ADC). The ADC on the ATTINY is well suited to this task as it has the ability to produce an internal 1.1V reference voltage regardless of the input voltage. In a traditionally referenced ADC the reference voltage would be the input voltage to the ADC and thus the read value would not change as the battery sagged.

The remaining four pins are used for SPI reprograming, controlling the constant current LED driver, and also driving a bidirectional LED that denotes the battery status. The battery status is indicated using a
bicolor LED that emits red light when driven in one direction and green light when driven in the other direction. By placing this LED between two IO pins it could be driven in either direction. The pins for the LED could be reused because some of the control lines are only used when changing the driver and are not needed to blink the light. Furthermore the function of these lines is dependent on another line, so the driver can be made to ignore them. The only artifact of this reuse is that the status LED may flicker briefly when starting the driver, but will otherwise show the correct battery status. Because the status LED draws 20mA, it was placed in series with a momentary push button so that it only turns on when needed. The circuit schematic for the beacon is provided in Figure 28, the board design in Figure 29, and the final Beacon is shown in Figure 30.

The code for the beacon is quite simple. It sets the LED driver to normal mode and then in a loop checks time against the time to perform the next action. The LED is set on and off by toggling one pin. As part of each loop the code also reads the battery voltage and based on preset thresholds sets the state of the two output pins that drive either side of the bidirectional status LED. The code can also flash a given binary code. In this case after the prescribed delay the driver writes the LED with the value of the next bit to send. The flash rate and byte code are configurable via compiler declarations at the top of the source code file. The source code for the beacon can be found at http://www.calvin.edu/academic/engineering/2013-14-team9/Code/TinyBeacon.ino.
Figure 28: Final Beacon Schematic

Figure 29: Final Beacon Layout
Figure 30: Final Beacon
8 Tracking Subsystem

The tracking subsystem takes the sensor’s data and determines the targets’ positions relative to the spotlight. Using the positions, it interfaces with the User Interface subsystem. It sends the User Interface the positions of the targets, and receives the desired target. The tracking system converts the position of the target to commands and sends them to the Actuation and Lighting systems.

8.1 Requirements

The design shall have a 1 foot accuracy in locating the target. The design shall have a latency equal to or less than 180ms from the time the video is received to the time the position of every target is output. The design shall work equally well in fully lit and blackout conditions. The design shall follow the target when the target can't be identified for 5 seconds.

8.2 Decision Criteria

The decision criteria for the subsystem are detection rate, development time, and cost. A high detection rate will ensure that users trust the system. Low development time and cost will be good stewardship of limited resources.

8.3 Decision Alternatives

The team has split the design into three components: algorithms, hardware, and software. Algorithms calculate the positions of the targets, and run on the software and hardware.

8.3.1 Algorithm

The problem of tracking an object from a video stream is an active area of research in the field of image processing and computer vision. The problem breaks down into finding a representation for the object being tracked, extracting features from the video to make detection possible, detecting the objects, and tracking the object with an algorithm.\textsuperscript{55}

The alternatives for feature extraction are color, texture, edge detection, and optical flow. Color locates features that match a certain color. Texture locates features that match a certain texture. Edge detection matches edges in the image. Optical flow locates features based on the displacement of pixels in the video.\textsuperscript{61}

The alternatives for object detection are point detection, background detection, segmentation, and supervised classification. Point detection detects points of interest that are useful for tracking.
Background detection detects objects that aren’t part of the background. Segmentation separates the image into segments that are visually similar. Supervised classification uses a machine learning algorithm to identify which regions match a dataset of positives.

The tracking was done with a point tracker, since the target is represented as a point. The three design alternatives are deterministic trackers, Kalman filters, and particle filters. Deterministic trackers work by finding the object that corresponds to the same object in the next frame. Kalman filters and particle filters use statistical methods to track the object. Kalman filters and particle filters are more effective than deterministic methods, since they account for error. Kalman filters assume the errors are Gaussian; particle filters do not.

8.3.2 Hardware

The team evaluated FPGAs, DSPs, GPUs, and CPUs. ASIC implementations were eliminated early for their high cost.

FPGAs provide high performance on embarrassingly parallel problems. They would provide the best performance of all the options, since the algorithms are largely parallel. However, FPGAs require a long development time. They lack niceties such as floating point and easy access to memory. Development happens at a low level of abstraction.

DSPs also require long development time. High performance code needs to be optimized specifically for the architecture to take advantage of parallelism. The architecture is also unfamiliar to the team and would require time to learn. The existing video libraries for DSPs do not provide the necessary algorithms and the team would have to develop some in-house. DSPs also have a high cost. A TI DSP evaluation module for video processing costs over $2000.

GPUs provide high performance similar to DSPs. GPUs require moderate development time. They are programmed with specialized languages or libraries, such as CUDA or OpenCL. GPUs have moderate cost. NVidia sells a development kit that costs 349 EUR.

CPUs provide moderate performance. CPUs require low development time, since the team is familiar with them already. CPUs have low cost, since customers can use existing PCs to run our system.
8.3.3 Software

The software was developed taking advantage of existing libraries. Writing our own library would have taken too much development time. For example, OpenCV represents the work of 800 person-years.\textsuperscript{72}

Alternative computer vision libraries were selected from open-source libraries. Open-source libraries deliver sufficient features at no cost. The choices for libraries included OpenCV, VXL, and LTI.\textsuperscript{73}

OpenCV had a low development time. Several team members were already familiar with OpenCV. OpenCV is also mature and well-documented online. The team’s industrial consultant also suggested the use of OpenCV.

VXL or LTI would have had a slightly larger development time. They were unfamiliar to the entire team, so learning to use one would have been part of the development process. Neither has as extensive documentation as OpenCV. OpenCV has the best performance, followed by VXL and LTI.

The alternatives for programming language were C, C++, and Python. The team chose these alternatives because they were languages the team already knew. Learning a new language would have increased the development time excessively. These languages are also ones that the computer vision libraries have bindings for. VXL and LTI support C++ and OpenCV supports C, C++, and Python.

All programming language alternatives had the same cost. All alternatives have no cost to use.

C had the best performance, followed by C++, and finally by Python. C++ performed within a factor of 2 of C.\textsuperscript{74} Python performed on average 36 times slower.\textsuperscript{75}

C would have required the most development time. It lacked abstractions for rapid development. C lacks object-oriented programming, and does not have convenient support for multidimensional arrays.
C++ required moderate development time. It provides useful abstractions such as object-oriented programming, higher-order functions, RAII, and polymorphism. These features ease the use of multidimensional arrays.

Python required the least development time. It supports object-oriented programming and dynamic typing. Multidimensional arrays are well supported with existing libraries.

8.4 Design decision

8.4.1 Hardware

The team chose the general purpose computer. Development time is the differentiating criterion. The general purpose computer also integrates well with existing infrastructure in theaters since theaters often have computers already.

8.4.2 Software

The decision criteria important for software were performance and development time. Performance was necessary to meet the requirement of less than 380ms latency. Minimal development time were necessary to stay on schedule.

The best option under these criteria was the middle ground of C++. C++ has nearly the best performance, and also has enough high-level features for fast development.

OpenCV was the best alternative for the computer vision library. It had the best performance and least development time.

8.4.3 Algorithm

The software for infrared tracking extracts features using the texture of the image. The IR beacon creates an easily identifiable point in the image. Color wasn't available since the camera operates in the infrared. Edge detection was not ideal, since the beacon doesn't have defined edges. Optical flow was more complicated and would have required more development time.
The object detection was done with a combination of background subtraction and a classifier. Segmentation did not map as obviously to the problem and so would have increased development time as well.

The tracking was done with a Kalman filter. The Kalman filter took less development time because it was already familiar to the team and was more effective than deterministic methods.

8.5 Implementation Details

The tracking subsystem was implemented following the design decisions described earlier. The code can be found at https://github.com/pfynan/spoor

The tracking subsystem breaks down into several components. Video enters the vision subsystem and is processed to estimate the location of the target. The estimated location is drawn on the video, and sent out to the GUI system. The estimate is also sent to the spotlight communication component to point the spotlight. The GUI communication component communicates with both the tracking subsystem and the spotlight communication subsystem. The components are integrated into a multithreaded architecture.

The spotlight communication subsystem takes commands from the other modules and sends them over TCP to the spotlight. The system uses the Boost Asio library to handle the TCP sockets.

The GUI communication subsystem is an Apache Thrift server autogenerated by Apache Thrift. Since it must constantly handle connections from the GUI, it runs in its own thread. It calls into the vision and light communication modules to promulgate the messages from the GUI.

The vision subsystem is where the bulk of the image processing happens. It is inspired by a paper on nighttime vehicle detection that uses a threshold, classifier, and Kalman filter. It receives video from the
sensing subsystem using the Gstreamer library. After feature extraction and tracking is done, it annotates the video and sends it to the GUI also using Gstreamer.

The vision subsystem performs both feature extraction and tracking concurrently. The subsystem tracks with a Kalman filter, which is an algorithm that estimates the position of the beacon from the measurements. The Kalman filter is defined with the state-space equations:

\[
\begin{align*}
    x_k &= F x_{k-1} + w \\
    z_k &= H x_k + v
\end{align*}
\]

Where \( x \) is the state, \( z \) is the measurement, \( w \) is Gaussian process noise, and \( v \) is Gaussian measurement noise. \( F \) is the state transition model, and \( H \) is the measurement model. Since the location of the beacon is measured directly, the measurement model is just an identity matrix. The state transition model is a constant velocity process in two dimensions. Constant velocity is reasonable since people tend to walk at a constant rate between two points. Thus, the state transition model is the matrix:

\[
F = \begin{bmatrix}
    1 & 0 & 1 & 0 \\
    0 & 1 & 0 & 1 \\
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1
\end{bmatrix}
\]

The system extracts features by first Gaussian blurring the video to remove noise. Then, the system performs a morphological closing to fill irregularities in the image. Next, the brightest parts are extracted with a simple threshold. After the video is separated by connected components into blobs (contiguous regions), each blob is evaluated for its similarity to the beacon.

Each blob is compared on the criteria of size, shape, and expected location. These parameters are determined by running the feature extraction code on a representative test video. The size is the area of the blob. The shape is measured with the first Hu moment. The expected location is obtained from the Kalman filter. However, the system determines if the location is consistent with the expected by using the method in a paper by James W. Davis. The system calculates the covariance of the predicted location using the Kalman filter, then calculates the Mahalanobis distance between the expected location and the actual location of the blob.

The Mahalanobis distance is a measure of distance that incorporates the covariance of the points being measured. It is defined as:

\[
d(x, y) = \sqrt{(x - y)^T S^{-1} (x - y)}
\]

Where \( x \) and \( y \) are points and \( S \) is the covariance matrix obtained from the Kalman filter.
The system rejects blobs that are inconsistent with 95% of the test video (described later). Of the remaining blobs, the one with the lowest Mahalanobis distance is chosen to update the Kalman filter and sent to the actuation system.

The Hamming distance between the expected blink pattern and the actual blink pattern is checked and the Kalman filter stops updating if the expected blink pattern doesn't match the actual blink pattern.

The team tested the spotlight systems each individually against test modules. The communication components were tested by sending data manually over TCP and inspecting the results. The vision component was tested on videos of a target walking with the beacon at walking speeds in zigzag patterns, since that pattern is the most likely to contain high acceleration. The test videos were used to set the parameters of the vision system. The vision system was also tested on additional videos to see if it met the requirements, including one in blackout and one fully lit by stage lights.

The entire subsystem was tested with test video while the communication to the spotlight was manually inspected. Finally, the subsystem was integrated into the entire system and tested to ensure it met all requirements.
9 User Interface Subsystem

The purpose of the user interface subsystem is to provide the non-technical user with a way of controlling the tracking and lighting functions of the system. This chapter covers the control path, the hardware used, and the design of the graphical user interface.

9.1 Research

The team conducted research into other graphical user interfaces used in theatre lighting. Some current computer controlled lighting systems have software that uses a USB-DMX interface. This allows the application to control any attached DMX enabled lighting fixtures. In particular, the team found a typical interface named SIUDI 8 that provides USB control of a DMX512 bus. This shows that a USB to DMX interface is possible and is utilized in current system. Many existing lighting boards also use a computer-based interface. This is especially useful to aid in system setup, but it is also used for show programming. These applications were used to assess how user interfaces were provided for existing systems and how our application could best fit into the existing environment.

9.2 Control Overview

The user interface will receive information from the operator in regards to the desired pointing position, tracking target, and light level. It will provide the user onscreen visual feedback. The user interface system will output the target tracking, and lighting parameters to the appropriate subsystems. In addition to this, it will also be able to change the light-operating mode and perform maintenance operations such as system tests. This interaction is shown in Figure 32.

![Figure 32: Operator Controls and Integration Connection Diagram](image)

9.3 Hardware

For the hardware of the User Interface (UI) system, the team wrote a program that can be run from the user’s laptop/desktop. The following specifications ensure that the programs can run smoothly without any lag. The team did not design the hardware but is simply stating the minimum specifications of a laptop/desktop that can be used for UI system.
9.3.1 Hardware Requirements
The software is designed to run well without significantly fast hardware. It should run on most modern laptop or desktop machines with an Ethernet connection, mouse, and keyboard.

9.3.2 Operating System
The UI can be compiled for all major operating systems such as Windows 7/8, Mac OS X, and all major versions of Linux, as the UI is written in C++ using QT Library that supports all major platforms. This gives the team the flexibility of moving the code to a different computer with a different OS without major hurdles.

9.4 Software Design

9.4.1 Programming Language
The team wrote the software for the user interface in C++ using the Qt-Creator IDE with the Qt Library developed by NOKIA. Other possible programming the team considered were C, Ruby, Java, and Python. Other possible development environments the team considered were Microsoft Visual Studio or Eclipse IDE. The team made the decision to use C++ based off of the programming knowledge of each team member. C++ is the programming language that all the members have the most experience with. Also, C++ was the final choice because C++ is one of the fastest programming languages in terms of software performance, as it is a mature compiled language.

9.4.2 Software Overview
The software consists of four main functional buttons: Get Video, Lighting Control, Tracking Control, and Actuation Monitoring. Each main function of the software has its own block thread to ensure that each process can execute immediately without waiting for another function to complete.

9.4.3 Communication
The GUI (Graphical User Interface) software communicates to the Tracking Subsystem via Apache Thrift. Apache Thrift is an interface definition language and binary communication protocol that is used to define and create services for numerous programming languages such as C, C++, Java, Python, etc. The GUI will have various functions to control the Actuation and Lighting systems via a communication channel with the Tracking software of the Tracking subsystem. Thrift includes a stack for creating client and servers as shown in Figure 33. “The top part is generated code from the Thrift definition. The services
generate from this file client and processor code. The protocol and transport layer are part of the runtime library. With Thrift, it is possible to define a service and change the protocol and transport without recompiling the code. Besides the client part, Thrift includes server infrastructure to tie protocols and transports together, like blocking, non-blocking, and multi-threaded servers. The underlying I/O part of the stack is implemented differently for different languages.87,88,89

The team chose Apache Thrift over other communication protocols because they wanted to develop a network service between the User Interface and the Tracking & Control Software.

The User Interface software functions as a client, and the Tracking & Control software functions as a server for this communication. A Thrift file was written to describe this communication. This file was compiled, and it automatically generated the code in the destination language. The destination language in this case is C++ since both of the software components were written in C++. The Thrift description file is shown in Figure 34 below.

This code is used to compile and generate several C++ files. The Tracking & Control software functions as a server to provide functions that can be called from the client, the User Interface software. In the “Service Tracking” section of the code, those functions are used to control the Tracking, Actuation and Lighting subsystems. Figure 35 shows the block diagram of the system communication including the communication between the User Interface and the Tracking Software components.
As shown in Figure 35, The Tracking Software is the central controller of other subsystems. The communication path of the system functions as follows.
• The User Interface calls a function from the Thrift generated code, and a message is sent to the Tracking software via the Apache Thrift protocol.

• The Tracking software sends a responsive message back to the User Interface as a confirmation.

• The Tracking software also sends the message to the actuation system via Ethernet using TCP/IP protocol defined in chapter 5.

• In the case of a lighting system command, the actuation system sends the message to the lighting subsystem over TTL serial.

As an example, to turn on or off the spotlight, the user interface will call the setOn/setOff function, and then the Tracking software will receive that message and send the message to the Actuation system. After that, knowing that function will control the spotlight, the actuation will send a corresponding TTL Serial message to the Lighting System. Figure 34 shows an example of the communication between the GUI and the Tracking Software.

In Figure 36, Tracking_server.skeleton is the executable compiled file from the generated Thrift communication codes. This file starts a communication protocol between the Tracking system and the User Interface system. The functions setMode and setOnOff were called as shown above to verify that the Tracking Software had received the messages and would execute the rest of the task.
9.4.4 User Interface Design

The ideal GUI would use a PC-based user interface. In the main screen there would be four main features: a clickable view of the stage, tracking target selection, light parameter adjustment, and actuation control. Since one of the main criteria for the software is a user-friendly interface, the video stream displaying the view of stage was fed to the main screen. A clickable view of stage is convenient for the operator to determine the pointing direction of the spotlight. Tracking mode selection switches between manual control and automated tracking.

![Figure 37: Preliminary Design 1](image)

The first preliminary design of the GUI is shown in Figure 35. The main window features four buttons and one video widget. Ideally, the video widget would display the video stream feeding from the camera connected to the Raspberry Pi computer board. The “Stream” button would begin the video stream from the camera. The “Track button” would turn on/off the tracking function of the Tracking software. The “Actuation Monitoring” would display the window on the right half of the figure that has the sliders & spinners to adjust the panning and tilting positions of the Actuation System. “Lighting Monitoring” would display the window on the left half of the figure that features “Lighting Toggle” button to turn on/off the spotlight and one pair of spinner & slider to adjust the brightness/intensity of the spotlight.

The first major hurdle was embedding the video stream into the GUI. Despite looking up the documentation and trying more libraries for video such as VLC, GStreamer, and FFMPEG, the video widget could not display the stream as intended. Thus, this widget was removed. The other major drawback was a lack of functionality in this preliminary design. Therefore, a second preliminary design was created that incorporated many changes to its interface.
In this design, shown in Figure 38, the video widget was removed. In the main menu of the GUI, the “Get Video” button was added in. This button displays the stream from the webcam of the Raspberry Pi. The video stream in the figure above was streamed using avconv software (an FFmpeg based audio/video encoder). The “Stream” button was used to test the video stream by plugging the webcam into the desktop instead of the Raspberry Pi computer. In the “Actuation Manual Control” window, four more buttons were added to this design. Their functionality is described below.

- Calibrate (void calibrate()): Sets the reference position of the mirror of the actuation system
- Halt (void halt()): Pauses the movement of the mirror
- Sleep (void sleep()): Puts the actuation system into a slow power mode
- Wake (void wake()): Resumes the actuation system from the low power sleep mode

This design was a significant improvement, but it also had a number of issues. The first issue was the lack of needed functionality. In the Tracking Control, the button “Manual/Automatic” could not toggle between Manual Tracking and Automatic Tracking modes. Also, there was no method to display the position of the mirror from the Actuation system in this design. Moreover, the avconv streaming method was not ideal. Therefore, the team decided to use GStreamer for the video streaming method (described in section 7.1). Following this change, the team had finalized the user interface shown in Figure 39 below.
In this design, all the windows were modified. The Tracking Control window is shown in Figure 40.

**Figure 40: Tracking Status Window**

In this box, there are two buttons to toggle each mode of the Tracking function from the Tracking Software.

- Automatic (void set_auto_Mode()): Sets the Tracking to automatic tracking mode
- Manual (void set_manual_Mode()): Sets the Tracking to manual tracking mode. To do manual tracking, user can click on the position of the video stream to direct the tracking pointer.

For the Actuation Control window shown in Figure 41, another button was added which is the “Get Position Button”.

95
The “Get Position” works from this following function

- Get Position (void getActualPos()): Displays the x, y coordinates of the mirror

After clicking this button, another display window will pop out and display in Figure 42.

In the Lighting Control Window, one major change was made. Two separate buttons replaced the “Lighting Toggle” button for turning on/off the spotlight to execute the function setOnOff.

- OFF (void set_off()): Turns the spotlight off
- ON (void set_on()): Turns the spotlight on

In the main menu, the “Stream” button was removed from the final design. To execute the streaming pipeline, the computer executes a remote control command (ssh command) to the raspberry Pi. The “Get Video” Button executes the following command to display the stream:

```
gst-launch-0.10 -e udpsrc port=5000 caps="application/x-rtp, " "media=video, clock-rate=90000, "
```
9.4.5 Design Outcomes

Most of the functions executed properly and integrated well with the rest of the system. However, there were some unwanted outcomes in this final design. In the lighting intensity adjustment function, the GUI and the Tracking Software protocol communicate with each other properly. However, the lighting system cannot execute that function to dim the light to substantially lower brightness. The system does not display the actuation coordinates properly due to the display format set in the codes of the User Interface.

Despite these difficulties, the team successfully demonstrated the user interface system along with the rest of the project during the Senior Design Open House.
10 Actuation Subsystem

The purpose of the actuation subsystem is to point the beam of light at the desired point on stage as directed by the tracking and control subsystem. This subsystem was primarily designed and built by team member Jake Roorda. Jake was chosen to lead this system based on his practical experience with robotic systems, motor control, and building small mechanical devices. This chapter will cover the major design decisions that were made in designing the actuation system and the implementation and testing of the final design.

10.1 Research

The team began their actuation research by establishing the actuation requirements. The team was able to learn about the typical usage of followspots through interviews with the Calvin Theatre Company (CTC) and the Calvin Office of Campus Events (OCCE).

CTC introduced the team to how they use spotlights and what their requirements for the system would be. Two of their major requirements were that the system be small enough to fit into their existing lighting area without being a distraction and that the system be controllable via DMX so that it could be scripted with the rest of the show on the lighting board. CTC also introduced the team to the lighting hardware they use and their general lighting procedures. This included the use of mid-sized ellipsoidal lights as spotlights, and a Rosco I-Cue mirror pointing system that they occasionally fit to these lights.93 94

The team also conducted research into current existing moving spotlight systems. While at least one tracking spotlight system has been brought to market in the past, none of the lights have caught on. The most successful of these tracking systems appears to be the Wybron Autopilot II. This system utilized a moving mirror to point the beam of an existing spotlight. With a list price of $33,850.00 not including the spotlight, the Autopilot was generally considered too expensive and too complicated for widespread use. The complexity was a result of their sensing system that required numerous sensors to be placed around the stage. Their system also needed a complex calibration procedure.95 96 97

While tracking lights have yet to go mainstream, motorized spotlights and other moving head lights without tracking are commonly used today, especially for large concerts and performances. Some of the first electric moving head lights were introduced as early as 1925.98 Today these lights form the core of many lighting shows and they can be easily spotted on TV shows like Dancing with the Stars as seen in Figure 43.
10.2 Preliminary Testing

The team’s first goal in designing the actuation system was quantifying how quickly the light would need to be moved. To accomplish this, the team conducted their own research on a variety of performance stages on campus. While one member moved quickly around on stage, a second member watched from the approximate position of a theoretical spotlight. Using a protractor and a stopwatch to measure their angular movement the team calculated the observed pan rates. These rates were doubled to allow for actors that were running instead of walking quickly and as a safety factor. The team determined that a pan angle of at least 120° was needed and that the system needed to be able to pan at least 60° per second.

Tilt rates could not be gathered as easily because at least one team member would need to be on the catwalk over the stage, so the team calculated those using estimated data. While the team could have pursued getting this data they knew the tilt speed would be less of a concern for the prototype system as it would be mounted in front on the stage and not above it. The worst can tilt rate comes when the spotlight is used to follow an actor from above and they walk directly from front to back on the stage. This is also the most extreme for lower lighting locations. The team established 20 feet as the lowest height that could be expected in common theatres. They also established a fast human walk at 6 feet per second.

The setup of this problem can be seen in Figure 4. The goal is to find the maximum rate of change for the spotlight angle Θ. Θ can be defined in terms of X using the equation

$$\theta(x) = \arctan\left(\frac{x}{20}\right).$$

The rate of change of Θ can thus be expressed by the time derivative of this statement.
\[ \frac{d\theta}{dt} = \frac{20}{x^2 + 400} \frac{dx}{dt}. \]

This equation is maximized when \( x = 0 \). At this point the equation becomes

\[ \frac{d\theta}{dt} = \frac{1}{20} \frac{dx}{dt}. \]

The time derivative of \( x \) is the walking speed in feet per second which was assumed to be 6 feet per second. This means the system shall be able to tilt at least 0.3 radians per second or \( 17.2^\circ \) per second.

![Light Tilt Calculation Setup](image)

**Figure 4.4: Light Tilt Calculation Setup**

The team found that an auxiliary requirement would press the need for a faster tilt rate. The team wanted to be able to replicate the roving spotlight effect where a spotlight is swung in seemingly random patterns about the theatre. To accomplish this, they deemed it necessary to move from the back of the stage to the front of the stage when mounted overhead in no less than 3 seconds. With a 50 foot deep stage and the light mounted 25 feet above the middle of the stage this requires moving 90° in three seconds. Thus this requirement pushes the maximum tilt rate to 30° per second.

### 10.3 Overall Design Alternatives

The primary design decision for the actuation system was how to point the light. Both direct pointing and mirror pointing can be used to point a spotlight. In direct pointing, the light is mounted to the lighting scaffold or batten pipe via a motorized yoke that can control and pan angle of the light. The light is hung at the bottom of the yoke near its center of mass such that a reasonably small motor mounted on one side of the yoke can tilt the spotlight. This setup is shown in Figure 4.5.
Alternatively, mirror-based pointing relies on a movable mirror to point the beam from a fixed light. Existing mirror-based pointing systems connect to a bracket on the end of most lights that is typically used to mount color media. These pointing systems usually have a support that is used to mount the mirror in front of the light. Two small motors can then be used to pan and tilt the mirror. This will pan and tilt the beam of light such that the function of the light is the same as moving the light itself. While not entirely intuitive, it is true that the beam of light that hits the stage off of a mirror will be identical to the beam of the light that would be created if the light was pointed directly at the stage. This type of system is shown in Figure 46.

The primary advantages of the direct pointing system are that it places no restrictions on the light’s beam size, it allows for 360° of pan and about 270° of tilt, and for some venues the motion of the light itself can
be an interesting part of the show. Direct pointing systems are able to have such a wide range of pointing angles because the only restriction is that they cannot point through their own base. Assuming that the base is 45° out from the light source the base only block 90° of tilt leaving 270° of usable tilt.

The major drawback is that the entire spotlight must be moved to point the light. Existing lights can easily weigh as much as 10kg whereas a mirror may average around 400mg. The mirror is also contained within about 6 inches while the entire light may be two or three feet long. This minimally 4 to 1 length difference is important because the moment of inertia is proportional to this length squared. When one combines conservative estimates of how these two factors affect torque it leads to a 160 times increase in needed motor torque to get the same performance. The team assumed that motor torque scales linearly with motor weight. This is reasonable because it is largely dictated by the number and size of the winding on the motor and the size of the magnets. This estimation was found valid through a sampling of motors from Maxon.

Existing mirror pointing systems use motors that are about 4cm cubed and weigh about 100g. A motor that is 160 as massive would weight 16kg and would be about 30cm cubed! Besides their size, these motors would also take more energy to move the light, as it follows naturally that a motor that is 160 times as powerful will in general need 160 times as much electrical power. These dimensions are clearly not reasonable, so existing directly pointed lights do sacrifice performance especially in terms of speed and acceleration and they usually stick to very small lights like pinspots in order to keep the motor size reasonable.

The direct pointing system also needs a lot of free air space to guarantee unimpeded movement. For a three foot long light this entails guaranteeing a three foot deep and three foot diameter cylinder of airspace. This often means that the light cannot be readily hidden. At a rock concert where the light is part of the show this is not a problem, but in theatre this would detract from the production.

The major advantage of mirror-pointing is that only a small mirror (4-12 inches square) needs to be moved in order to move the beam of light. This means that small motors that may only be an inch or two in diameter and 50-200g can match or exceed the performance of even very powerful direct pointing systems.

When a beam of light hits a mirror at an angle the light is reflected doubling the angle of the reflected beam as shown in Figure 47. This doubling makes it so that the mirror only needs to tilt half as much as the desired lighting tilt angle. The first major downside of mirror based pointing is that the mirror can contribute distortion and dirt spots to the light. Additionally, if the mirror only reflects a portion of the light beam it can distort the shape of the light and make it very hard to focus the beam. Finally, the use of
even an ideal mirror system limits the maximum tilt angle to no more than 180° as shown in Figure 48. Any additional tilt would require the light to bounce off the back of the mirror and this is where the connection to the motors shall be. In reality the 180° maximum cannot be obtained because in positions near 0° the spotlight blocks the reflected light, and near 180° the mirror would have to be very large to continue reflecting the entire beam. Nevertheless, 180° is still a useful theoretical metric for comparison purposes.

![Figure 47: Mirror Doubling the Angle of the Reflected Beam](image)

10.3.1 Decision

The team chose to use mirror-based pointing for a few reasons. The first reason comes from a consideration of the customer. The team is primarily targeting smaller theatre venues with their prototype
system. For use in a small theatre the additional size of a directly pointed spotlight is a significant issue while the limited pointing range of a mirror-based system is not a large concern. The second reason comes from the decreased cost of the prototype system due to the cost savings of smaller motors and smaller motor controllers. This is important due to the team’s tight budgetary constraints.

10.3.2 Implementation

The team also believes that by modifying the standard mirror-pointing system's design a better alternative could be created. Most existing mirror pointing systems use a small elliptical mirror that only reflects a portion of the light.\textsuperscript{110} As mentioned earlier, this makes the light difficult to focus. If the mirror is replaced with one that is approximately 70\% larger this can be avoided for the needed tilt angles. This is shown in Figure 49. Trigonometry was used to show that to reach a 60° angle with no light spillage the mirror needs to be twice the size of the light beam. This can be shown by solving the relation

\[
\text{Length} = L \cdot \cos^{-1}(60°).
\]

Similarly the pan angle can be expanded to a full 360° if the entire frame that supports the mirror is rotated instead of the simply rotating the mirror. In existing systems, rotating the mirror 360° could be done, but the support that holds the mirror will block the light. If the entire frame is rotated with the mirror it will no longer block the light. Both of these changes will increase the size and weight of the
component that needs to be moved, but the team determined that this trade off was justified given the added performance provided by the modifications.

10.4 Actuation Hardware

In robotics, an actuator is a motor system responsible for moving part of the system. In this case the actuation system is responsible for pointing the beam of light. There is a considerable amount of hardware that needs to come together to make the actuation subsystem. The largest portion of this hardware is the frame that holds the mirror. The motors that drive the mirror are the next largest portion of this hardware. Other pieces include the mirror itself and the electronic hardware to control the motors.

10.4.1 Frame Design

The actuation hardware is all built around a frame that attaches to the lighting assembly and allows the mirror to pan and tilt robotically. This design was built around the main bearing. In order to pan the light the mirror needed to rotate along the axis of the center of the light. This meant that the light needed to shine through the center of the bearing. After much searching the team was able to find a lazy-susan type bearing online that had a center diameter large enough for the light to shine through at a reasonable cost.

10.4.2 Power Transmission

To transfer power to this bearing the team considered using fixed gears, using a contact wheel, or using a V-belt. Fixed gears were attractive because they would not slip and would be easy to assemble, but they presented two issues. First, finding a fixed gear large enough to encircle the light was a significant challenge, and second, this gear would introduce backlash into the system which could make it hard to keep the light pointed exactly where it was needed without it bouncing back and forth between the two nearest gear teeth. The team also considered a contact wheel, but decided against it because of the possibility that it could slip and introduce error into the pointing.

The team finally decided to use a V-belt to transfer the power. V-belt pulleys were available for most of the needed sizes, though the team did have to make their own pulley that went around the light itself. Thanks to the help of shop supervisor Phil Jasperse, the team was able to make this pulley on a lathe and connect it to the rest of the system. While V-belts can also slip, they can be tensioned such that slippage is not likely for the needed torques and they avoid the backlash of a geared drivetrain. While backlash will be present in the geared stages closer to the motor, this backlash will be scaled down by the mechanical advantage provided by the final belt drive. The final V-belt pulley can be seen in Figure 50.
For much the same reasons the team used a belt drive to make the final linkage between the tilt motor and the mirror assembly. In this case both of the required pulley could be acquired as off-the-shelf components. The tilt motor and pulley are shown in Figure 51 and the pan motor and pulley are shown in Figure 52.
Many custom components were built. Two of the most complex components were the bearing supports and the mirror bearing support. CAD drawings for these parts are shown in Figures 53 and 54 respectively.
10.4.3 Supports

The supports that connect the bearing to the mirror and pan motor are in general made from scrap aluminum from the Calvin metal shop. Aluminum was chosen for its low density, machinability, and strength. The frame is made in pieces that bolt together using tapped holes and Standard English machine screws. This type of system was chosen over a welded frame because it allowed for individual components to be easily changed and it allows the system to be easily disassembled and reassembled during building and testing.

The part designs were modeled using Autodesk Inventor to assess how the entire assembly would come together. This was important to guarantee proper part clearances, and the team went through dozens of digital design revisions before a single component was produced. Many parts of the assembly such as the gearing, bearings, and shafting was ordered off-the-shelf, while others like the supports and the bearing mounts had to be custom made in the Calvin College metal shop. The Inventor Model is shown in Figure 55 while the final assembly can be seen in Figure 56 and Figure 57.
Figure 195: Autodesk Inventor CAD Model

Figure 206: Complete Actuation System
Another design decision was the type of motors that would be used. To meet the requirements the motors need to be able to allow for fine control of the mirror within $1/4^\circ$ to maintain a pointing resolution of 6 inches from a distance of 50 feet. This is because the $\arctan(0.5/50)$ is approximately $0.25^\circ$. Other requirements these motors must meet include that these motors shall be able to move at the desired rates of 60 and 30 degrees per seconds for the pan and tilt axes respectively, and that they shall run less than 30dBA from 10 feet to meet the noise requirements.

One choice that is used quite frequently in industry is the stepper motor. These would allow for accurate open loop control of the mirror. Without additional gearing, a stepper motor with 400 steps per revolution and a 16 level micro-stepping driver could be used to point the mirror in .06$^\circ$ increments.$^{112}$ The other main benefit of stepper motors is that when they are not moving they actively hold their current position with a considerable amount of torque. While these aspects are nice, they do come at a cost. Stepper motors are relatively slow, and if the motor skips a step due to the needed torque exceeding the available torque, the open loop control is broken. In addition, the stepping of stepper motors can make them quite loud such that careful vibration deadening would have to be used to quiet the system. Stepper motors also use power to simply hold the mirror in one place, and in fact it is in that
condition where they draw the most power. This would reduce the overall efficiency of the design significantly because the mirror would often be stationary.

The other main choice for the mirror actuation motors are DC gear motors. Like stepper motors, these come in a variety of sizes, but they also can be found at different rotational speeds thanks to a variety of gearbox options. Gear motors by themselves have no form of feedback, so unlike stepper motors they cannot be used to make an accurate open-loop system. Instead, some form of feedback would be necessary to make a closed-loop motor controller. Both potentiometers and encoders could be used to sense the rotation of the motor and provide feedback. When compared with stepper motors, small DC gear motors are generally faster, running in the 5000-10000 rpm instead of the 100-1000 rmp typical of stepper motors. They are also generally quieter due to naturally constant motion instead of stepped motion. They are more efficient, but as noted they are harder to position accurately and cannot actively hold the mirror motionless.

Finally, brushless DC motors and three phase motors are two additional options for mirror pointing. These motors are both controlled by a three phase AC voltage driven by a special driver circuit that looks at the state of the motor either via an encoder or feedback from the voltage lines in order to drive the correct voltage levels to spin the motor. Because of this built in feedback, they are a natural choice for a closed-loop system. They also avoid some of the torque dead zones that exist in brushed DC motors.

The major deterrent of these motors is their cost. While they predominate the market for small hobby vehicles, larger and slower spinning variants are quite expensive, as are the electronics required to drive them. For example the size motor needed for the pan system would typically run at or above $200 used. Given that the target distributor cost of the theoretical production system is $2500 it would be reasonable to use a BLDC motor for the pan motor in the final system given that at quantity the cost could be further reduced.

Because of the cost, the team did not expect to be able to use these motors in the prototype design. Fortunately, upon searching through some spare parts the team found a nice brushless DC motor shown in Figure 58. This motor uses hall-effect encoders to sense the position of the rotor. Because this motor was quite large and powerful, the team elected to use it for the pan system which demanded a larger motor than the tilt system.
In contrast to the pan motor, the team elected to purchase the motor that would serve to tilt the mirror. This motor could be much smaller than the pan motor and thus did not represent a large budgetary pressure. Based on personal experience with previous robotics project, team member Jake Roorda chose to use a metal brushed DC gearmotor from Pololu Robotics & Electronics. The particular selected motor provided most of the required gearing in its gearhead and came with a built-in Hall Effect encoder. These were both significant benefits as the ideal output speed of 10 rpm would be difficult to achieve with external gearing, and the closed loop controller needed an encoder to close the feedback loop. The chosen motor is shown in Figure 59.

![Figure 59: Tilt DC Gearmotor](image)

10.4.5 Mirror

The mirror itself is the key piece of hardware in the actuation system. The team elected to use a flat mirror. This means that the mirror will not affect the final beam other than by changing its direction. A concave or convex mirror could be used to focus the beam in addition to reflecting it. The team did not seriously consider this option because a curved mirror would also alter the focus when it was moved in
addition to changing the beam angle. Initial design iterations placed the size of this mirror around 10 inches square. This followed naturally from an estimated beam size of 5 inches and the calculations shown earlier in this chapter.

Unfortunately, the team found out when building the spotlight that they had made a poor assumption in their initial calculations. The initial calculations ignored the effect of the 15° divergent beam angle. When this angle is brought into play it significantly increases the needed size of the mirror as the beam quickly grows larger than the initial 5 inches. The prototype mirror was set a 12 inches square as this was the largest size the team felt they could move reasonably. This mirror is also mounted 8 inches closer to the light assembly to prevent the light from growing in size too much before it is reflected. These design changes are evident when comparing the initial CAD model shown in Figure 19 and the prototype light in Figure 20.

In a production design this mirror should be a first surface mirror. In a first surface mirror the silvered mirror surface is placed on the front of the first pane of glass. In a traditional mirror it would be placed behind another pane of glass for protection. If a second surface (traditional) mirror were used the double reflection could result in a second beam edge. This extra edge would detract from the function of the spotlight. The prototype system does use a second surface mirror to save on cost and make the system more durable for testing, but any production system would certainly use a first surface mirror.

**10.4.6 Electronic Hardware**

In comparison to the mechanical hardware the electrical hardware is much smaller and more contained to given locations on the device, but it too plays a large role in the operation of the actuation system. The following sections detail the primary pieces of electrical hardware.

**10.4.6.1 Microcontroller Options**

The overall electrical configuration of the system is largely dictated by the layout of the device. Electronics are needed to control both the pan and tilt motor and talk to the main control system, but between the pan and tilt system is a rotating gap that is costly to cross. Consequently, in order to limit the number of connections that must cross the rotating to fixed gap the team elected to place a microcontroller on either side of the gap to control the local functions. To simplify development the team decided that the same microcontroller should be used on both sides. The microcontroller options the team considered are summarized in Table 11 below.
Table 11: Actuation Microcontroller Options

<table>
<thead>
<tr>
<th>Development Board</th>
<th>Microcontroller</th>
<th>Cost</th>
<th>Ethernet</th>
<th>GPIO Pins</th>
<th>IO Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino</td>
<td>ATMEGA 328</td>
<td>$25.00</td>
<td>No, Add-ons Available</td>
<td>20</td>
<td>5V</td>
</tr>
<tr>
<td>Prototyped ATMEGA</td>
<td>ATMEGA 328</td>
<td>$5.00</td>
<td>No, Add-ons Available</td>
<td>20</td>
<td>5V</td>
</tr>
<tr>
<td>TI Launchpad</td>
<td>TI MSP4300</td>
<td>$13.00</td>
<td>No, Serial Addons Could be Adapted</td>
<td>35</td>
<td>3.3V</td>
</tr>
<tr>
<td>Intel Galileo</td>
<td>Quark X1000</td>
<td>$69.00 [mouser.com]</td>
<td>Yes</td>
<td>20</td>
<td>3.3V</td>
</tr>
<tr>
<td>Raspberry Pi</td>
<td>Broadcom BCM2835, ARM11 700MHz</td>
<td>$35.00</td>
<td>Yes</td>
<td>25</td>
<td>3.3V</td>
</tr>
<tr>
<td>BeagleBone Black</td>
<td>ARM Cortex-A8</td>
<td>$45.00</td>
<td>Yes</td>
<td>~46</td>
<td>3.3V</td>
</tr>
<tr>
<td>Olimex LPC2478-STK</td>
<td>LPC2478, ARM7</td>
<td>Have, $135.00</td>
<td>Yes</td>
<td>40</td>
<td>3.3V</td>
</tr>
<tr>
<td>Olimex LPC-H2294</td>
<td>LPC2294, ARM7</td>
<td>Have, $70.00</td>
<td>Yes</td>
<td>20</td>
<td>3.3V</td>
</tr>
<tr>
<td>pcDuino</td>
<td>ARM Cortex-A8</td>
<td>$60.00</td>
<td>Yes</td>
<td>20</td>
<td>5V</td>
</tr>
<tr>
<td>NIOS II on DE2</td>
<td>Cyclone II FPGA</td>
<td>Have DE2, $269.00</td>
<td>Yes</td>
<td>80</td>
<td>3.3V</td>
</tr>
<tr>
<td>Papilio One</td>
<td>Xilinx XC3S500E FPGA, AVR8 Softcore</td>
<td>$65.00</td>
<td>No</td>
<td>48</td>
<td>3.3V</td>
</tr>
</tbody>
</table>

There were a few key things that the team looked for in the microcontroller. One of the more important criteria is the number of general purpose input and output (GPIO) ports and other built in communication busses. The team gave a slight preference to 5 volt I/O over 3.3 volt I/O on the assumption that more of the hardware would be at 5V levels. Because at the time the final circuit was not set in stone I2C, SPI, and TTL Serial were also considered requirements in case they are needed. Every one of the devices the team considered provided these interfaces, so the scoring that item came down to how easy they were to implement in software.

Another feature that the team looked for was built in Ethernet. This is scored differently between different microcontrollers, on account that some microcontrollers have the Ethernet hardware, but leave it up to the user to figure out how to use it from the component data sheets while others provide fully featured libraries that can be used to start communicating over Ethernet with very little development effort.
The team also expanded the microcontroller search to include devices without built in Ethernet. These devices would need an external Ethernet controller. Fortunately, a variety of Ethernet devices are available that use some form of serial communications to communicate with a microcontroller.\textsuperscript{126}

Processing speed was not a major concern. The communications updates will occur at approximately 30Hz to match the frame rates, and the motor control should not require an undue amount of processing power, so almost any modern microcontroller can be used to accomplish these tasks.

The last of the judging criterion was the approximate development time each microcontroller would need. This varies significantly among the options for a variety of reasons. Some microcontrollers have very good libraries, development environments, and available sample code all of which help to speed development while others do not. The basis of these assessments is shown in Table 82.
### Table 82: Microcontroller Development Time

<table>
<thead>
<tr>
<th>Microcontroller</th>
<th>Ethernet Dev. Hours</th>
<th>Motor Control Dev. Hours</th>
<th>General Dev Hours</th>
<th>Development Hours Est.</th>
<th>Uncertainty Factor</th>
<th>Dev Time Score</th>
<th>Source and Reasoning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino / Prototyped ATMEGA328</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>11</td>
<td>40%</td>
<td>10</td>
<td>Team member Jake has significant experience with this platform and is confident estimating the design time. Libraries are available for motor control and ethernet.</td>
</tr>
<tr>
<td>TI Launchpad</td>
<td>10</td>
<td>5</td>
<td>5</td>
<td>20</td>
<td>60%</td>
<td>5</td>
<td>This is a new platform to the team. Documentation exists, but it is quite thin in places. Motor control is baked into the microcontroller but can only be used with complicated IDE and configuration. External Ethernet would require porting the Arduino library.</td>
</tr>
<tr>
<td>Intel Galileo</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>12</td>
<td>50%</td>
<td>8</td>
<td>This platform is quite mature and well documented. Ethernet and motor control libraries are easy to find and use. The added complication of finding a linux distribution and configuring the board to run on its own does present some extra development overhead.</td>
</tr>
<tr>
<td>Raspberry Pi</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>12</td>
<td>40%</td>
<td>9</td>
<td>This development board has good built in ethernet support. Motor control libraries may need to be written, but good documentation exists. Again some overhead exists in terms of configuring the linux distribution</td>
</tr>
<tr>
<td>BeagleBone</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>11</td>
<td>60%</td>
<td>9</td>
<td>The cool development board has a built in touchscreen and runs linux. Team member Jake has already spent approximately 40 hours last summer trying to get a Linux distribution onto the board and made relatively little progress. The likely solution to installing Linux would be to cross-compile uC-Linux for the device, and this would take a substantial amount of time. Few drivers exist for the hardware in Linux, so this could complicate the ethernet connection as well.</td>
</tr>
<tr>
<td>Olimex LPC2478-STK</td>
<td>8</td>
<td>8</td>
<td>30</td>
<td>300%</td>
<td>1</td>
<td>46</td>
<td>The development board does not run Linux, but it would still require learning a new language and IDE. The board is supported only with schematics and a few notes and examples. Ethernet drivers would need to be written from scratch using the schematic and datasheets provided.</td>
</tr>
<tr>
<td>Olimex LPC-H2294</td>
<td>10</td>
<td>5</td>
<td>15</td>
<td>30</td>
<td>100%</td>
<td>30</td>
<td>This platform is quite similar to the Raspberry Pi, but it does not have the same extensive community and code examples online. That said, it does support an Arduino style language that would simplify motor driving code development.</td>
</tr>
<tr>
<td>pcDuino</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>12</td>
<td>60%</td>
<td>8</td>
<td>This FPGA development board uses a Xilinx FPGA, so it would require the team to learn Xilinx's development tools. This adds significant overhead. The softcore processor it can include is compatible to some degree with the Arduino environment, so this could make many Arduino libraries usable. Documentation is good though the user community is quite small.</td>
</tr>
<tr>
<td>NIOS II on DE2</td>
<td>15</td>
<td>12</td>
<td>3</td>
<td>30</td>
<td>50%</td>
<td>3</td>
<td>From previous experience with the DE2 board the team believes that ethernet could be a large challenge as other classmate's attempts to communicate over ethernet have not been successful. They also feel that while motor control would be possible it could take a long time to develop the code from scratch.</td>
</tr>
<tr>
<td>Papilio One</td>
<td>10</td>
<td>7</td>
<td>10</td>
<td>27</td>
<td>80%</td>
<td>4</td>
<td>This FPGA development board uses a Xilinx FPGA, so it would require the team to learn Xilinx's development tools. This adds significant overhead. The softcore processor it can include is compatible to some degree with the Arduino environment, so this could make many Arduino libraries usable. Documentation is good though the user community is quite small.</td>
</tr>
</tbody>
</table>
The final microcontroller decision matrix is shown in Figure 60. Each development criteria was given a weight on a 10 point scale with 10 being the most important. Then each microcontroller option was rated relative to each criteria with 10 points again being the best score. These scores are multiplied by the category weights and summed for each choice. The option with the highest score was selected.

![Figure 60: Final Microcontroller Decision Matrix](image)

The chosen microcontroller option was the prototyped ATMEGA328. This is the microcontroller used on most Arduino boards. It has all of the required features with the exception of Ethernet and team member Jake Roorda has extensive experience with the Arduino language and is a regular commenter on the Arduino forums and is thus able to develop new applications with it quickly. This microcontroller is also the least expensive option because Jake has one or two development boards for this chip that can be used for development. He has also built boards for it before from parts and has spare parts from that endeavor sufficient to build another board at no additional cost.

Because the ATMEGA328 does not have built in Ethernet an Ethernet controller also needed to be used. The team identified an Ethernet to SPI board sold by Hobby King for under $7.00. This board has supporting libraries and sample code that made implementing Ethernet relatively simple. During implementation, the team found that the example code did not exactly cover the intended use of raw TCP/IP packets, but the team was able to make the board perform as desired. More details regarding this interface are presented in the communications chapter. When compared to the traditional Arduino Ethernet Shield, this board functions at a lower level of abstraction with the microcontroller handling more of the packet formation, buffering, and decoding. This places more demands on the microcontroller and user code, but it allows for more complex features such as true DHCP and low level control.

10.4.6.2 Motor Position Sensing

The position of the mirror needs to be known by the microcontroller in order to move the mirror to the desired absolute location. The system’s sensors need to give an accurate absolute measurement so the system knows its initial state on startup, and they also need to give accurate measurements of a change in position so that the feedback structure can work well.
The choice of sensors was largely dependent on the exact motor choice and was left to the second semester. The two main options the team considered were potentiometers and encoders. Potentiometers are resistors that change resistance based on the rotary position of the shaft. They would be a natural choice for absolute position sensing on startup, but they may be too noisy to be used exclusively for PID control. If this is the case, they could be paired with an encoder. Encoders measure degrees of rotation and the direction of rotation, but most do not supply an absolute positioning reference.

In the prototype system, the team was able to use the integrated encoders in both motors to provide for relative motion sensing. For startup calibration, the system uses optical reflectance sensors to sense the rotational endpoints and map them to their associated encoder positions. These optical sensors are also positioned to function as limit sensors that will prevent the rotational joints from driving past their design limits. The tilt sensor, shown in Figure 61, senses the painted white band on the mirror pulley to know when to reset the encoder counter for the motor.

![Figure 61: Tilt Reflectance Calibration Sensor Assembly](image)

10.4.6.3 Motor Drivers

Two different motor drivers are needed to control the motors that will pan and tilt the mirror. The design uses a brushed DC gear-motor to tilt the mirror. This motor is controlled using an H-bridge circuit. These circuits, as pictured in Figure 62, allow high currents to be switched through a motor in both directions. This allows the motor to be run bi-directionally. A microcontroller with pulse-width

118
modulation support can be used to drive the H-bridge controller in order to gain speed control of the motor.

![H-bridge Motor Controller Generalized Schematic](image)

**Figure 62: H-bridge Motor Controller Generalized Schematic**

Team member Jake Roorda has in the past constructed his own working H-bridge circuits from discrete MOSFETS and BJTs, but he has since learned that H-bridges can be bought in one integrated circuit. A variety of H-bridge chips and breakout boards are available. The team elected to use a Freescale MC33887 motor driver IC on a breakout board sold by Pololu to control the tilt motor. The MC33887 motor driver IC or a similar model would also be used in the production model as it can easily handle the 3A of needed drive current and it provides a simple way to drive this motor bidirectionally for a low part cost. This motor controller was a spare from a previous project and its voltage and current ratings were sufficiently high to power the selected motor. This driver also provides a direct current measurement port that the team can use instead of adding a separate current sense resistor. By reusing this motor driver the team was able to be good stewards of both the monetary resources they had been given and also of the raw materials and energy that went into producing the motor driver.

Trust and Humility are two of the other design norms that influenced the choice of motor driver and implementation. For the user to be able to place their trust in the actuation system it must be safe. One way to make the system safer is to add the hardware necessary for the actuation microcontroller to sense motor over-currents. DC motors draw much higher than normal currents when they are stalled and not moving. This higher current can be an indication of trouble. It could mean that the mirror has become
stuck or more seriously that someone’s hand was caught in the system. Upon detecting this overcurrent the system shall cut power to the motor and alert the operator of the issue.

For the BLDC pan motor to be used, it needed to have a special three phase motor controller. These controllers resemble an H-bridge controller with an extra set of inputs and switching MOSFETS. Premade drivers that utilize the hall-effect encoders to sense the rotor position can be bought for anywhere between $50 and $600 depending on functionality and target market. After looking into the function of these motor controllers, the team decided that because they already have an onboard microcontroller they could implement a brushless motor controller for under $10. The team was able to obtain free samples of the TI DRV8313PWPR triple-half-bridge IC that was suitable for this application from Texas Instruments. While a variety of three phase driver chips were available and equally suitable for the task this particular driver could be quickly sampled for free from Texas Instruments.

Despite the electrical suitability of the IC sampled from TI it did present some physical challenges. The chip is in an HTSSOP package. The team looked for DIP alternatives to aid in simple prototyping but could not find any with suitable performance at a good price point. Fortunately, the team was able to find a premade HTSSOP breakout sold by OSH Park. This board broke out the controller pins to a much more convenient tenth inch spacing. The team was able to manually solder the chip to the breakout board and use thermal paste to connect the covered thermal pad. This assembly is shown in Figure 63.

![Figure 63: BLDC Motor Driver on Breakout Board](image)

The team was then able to use an ATMEGA 328 microcontroller to read the three encoders on the motor and map them to the appropriate driver actions. This is necessary because BLDC motors must be electrically commutated so that the motor windings produce the correct magnetic fields in relation to the rotor. The microcontroller can then use the state diagram described by
Table 13 to map the encoder readings to the correct output states of the three half-bridge drivers in the driver IC. The values in Table 13 were derived from an online literature search of motor driving tutorials from Texas Instruments, Maxon, and others.\textsuperscript{132,133}

<table>
<thead>
<tr>
<th>Enc. A</th>
<th>Enc. B</th>
<th>Enc. C</th>
<th>Phase 1</th>
<th>Phase 2</th>
<th>Phase 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>V+</td>
<td>NC</td>
<td>GND</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>V+</td>
<td>GND</td>
<td>NC</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>NC</td>
<td>GND</td>
<td>V+</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>GND</td>
<td>NC</td>
<td>V+</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>GND</td>
<td>V+</td>
<td>NC</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>NC</td>
<td>V+</td>
<td>GND</td>
</tr>
</tbody>
</table>

In practice this worked well. Initial tests showed that the motor could be controlled with accuracy down to +/- one encoder count or 1/6 motor revolution. These tests also showed that the motor performed reliably within the needed specifications at voltages of 12V and above. The motor itself could be driven up to 48V if needed for added torque and speed.

10.4.6.4 Other Electronics

While the previous sections have touched on the core hardware, there is a lot of associated support hardware needed to connect and provide for the core components. Each of the ATMEGA 328 microcontrollers was prototyped on a piece of perforated board along with the peripheral components. Because the design was still relatively fluid and because it consisted of mostly point to point wiring the team elected to use perforated board instead of etching a printed circuit board. The overall layout of the pan and tilt control boards is shown in Figure 64 and Figure 65. Figure 66 and Figure 67 show the completed boards.

Custom boards integrating all of the functionality provided by the plug in modules would likely be about half the size of the given boards. The tilt board component cost would run around $20.00 with the 2 layer PCB. The pan board would cost about $50.00 including a four layer PCB to make the needed connections. The bills of materials for these boards can be found at http://www.calvin.edu/academic/engineering/2013-14-team9/links/PanBillOfMaterials.pdf for the pan controller and http://www.calvin.edu/academic/engineering/2013-14-team9/links/TiltBillOfMaterials.pdf for the tilt controller.
Figure 64: Pan Control Board Layout

Figure 65: Tilt Control Board Layout
Figure 66: Pan Control Board

Figure 67: Tilt Control Board
10.5 Actuation Software

The primary purpose of the actuation software is to translate the commands from the main control subsystem and feedback data from the pointing hardware into commands that will drive the motors as near as possible to the desired position. The other tasks that are also handled by the actuation software are passing commands to the lighting system, reporting error conditions to the main control system, and sending out motor control signals. The motor control signals that control each phase of the bridge drivers.

![Diagram of Actuation Tilt Program Flow](image)

**Figure 68: Actuation Tilt Program Flow**

The overall sequential loop flow of the tilt program is shown in Figure 68. Each block in this diagram is explained in more detail in the following section. This plan shows a polling based program for all functions except motor encoder reading. The main advantage of polling based programs is that they typically take less time to develop. Most microcontrollers also only have a limited number of interrupt capable inputs, so it makes sense to save them until it is clear where they are needed. Interrupt based systems are typically faster because there is not the additional burden of repeatedly checking the state of an input. The team chose to use a polling based design primarily to save development time. This allowed
the team to get to a working prototype sooner. At integration time it was found that the encoders were missing reads at higher motor speeds, so the system was changed to read the quadrature encoder via the two available interrupt pins.

The code for the tilt program can be found at http://www.calvin.edu/academic/engineering/2013-14-team9/Code/tiltDriverR5.ino. This code uses the functions readEncoderA and readEncoderB to read the encoders as triggered by a pin change interrupt. The setup function runs once through initialization code including setting the input or output of the pins and setting up the serial port. The loop function runs repeatedly to check for and parse incoming serial commands and run the motor PID controller.

The functional block diagram of the pan code is shown in Figure 69.

Figure 69: Actuation Pan Program Flow

The pan code can be found at http://www.calvin.edu/academic/engineering/2013-14-team9/Code/panDriverR44.ino. It uses polling to sense the motor encoders. In the main looping function, once every 100 times the Ethernet is checked for new messages and if any messages were received they are decoded and run. Every time through the loop the motor encoders are read and the correct signals are sent to the motor controller. The EtherCard library was used to talk to the Ethernet
board, the SoftwareSerial library was used to connect to the light and tilt systems, and the PID library was used to calculate the motor signals.

10.6 Communications

The actuation subsystem communicates with the tracking system over Ethernet. This design decision is documented in the System Architecture chapter and the interface is described in more detail in the communications chapter.

As the actuation system is the only remote system with an Ethernet connection it is also tasked with passing along messages to the lighting system and to the remote half of the actuation system. The requirements for this bus were for bidirectional communication and a single signal wire. The single wire requirement allowed the rotating to fixed connection to be made by a three conductor cable that would carry power, ground, and a signal line. This allowed for the use of a coiled headphone cable. The team would have preferred a four connector coiled cable, but no COTS alternatives could be found.

The team investigated both Dallas One-Wire and a custom multiplexed serial bus for the connection. Dallas one-wire is a proprietary bus made to communicate with the manufacturer’s line of one-wire sensors. The bus is designed to send short messages back and forth between a master and slave device and it allows the slave device to draw power parasitically off of the data line. In contrast, the multiplexed serial bus would use the TTL-level serial functionality common to most microcontrollers and use a preset call and response scheme to multiplex and transmit and receive lines onto one bus. This method was suggested by a prominent member of the Arduino forum. In its original form it would have used external multiplexing hardware, but because the pan microcontroller did not have enough free IO pins the team found a way to accomplish the same effect though an implementation of the software serial library.

The team decided to use multiplexed serial instead of one-wire because the multiplexed serial standard was much simpler to develop. While Arduino libraries existed for both the master and slave side of the one-wire interface, the libraries were meant to work with or emulate certain sensors. Because the spotlight system messages are not compatible with temperature messages it would have taken the team 10-20 hours to alter the libraries and the team suspected that the end result may not be very intuitive or graceful. In contrast, the multiplexed serial method is quite intuitive with the only downside being a dictated code structure. In early tests, this bus worked well enough that the team decided to use it for the connection to the lighting system as well instead of the planned i2c bus. This again simplifies the needed wiring and reduces the number of digital IO pins needed to implement the system.
As stated in the System Architecture and Lighting chapters the Actuation subsystem is also responsible for relaying messages to the lighting system over an SPI bus. In these cases the message is relayed directly once it is recognized that the message pertains to the lighting system.

During the final system testing, the team found that the multiplexed system did not work well. It was very susceptible to noise from the switching regulators, motors, and motor drivers. This is likely because the bus is left undriven between sends from the master and slave system. The team tried to correct this situation with a pull-up resistor and small capacitance. In the end this helped improve the bus to the degree that it could be used for debugging, but it did not have the necessary reliability for the final system. Because of this, in the final system the multiplexed system was changed to a unidirectional TTL serial connection. This can use the same library and protocol, but no messages are passed back over the bus, so the bus remains driven at all times and free from interference.

### 10.7 Testing and Debug

As the team moved forward, a series of intermediate tests were used as milestones towards the final goal. Each of the tests listed below tests part of the overall functionality. The milestone’s number dictates timeline precedence. Steps 1-6 can be completed in any order or in parallel.

**Actuation Testing Milestone 1:** Receive Ethernet communications from the host computer

Test: The microcontroller on a development board shall be connected to its Ethernet hardware which will be connected to a network containing the test PC. The terminal application ncat shall be used on the testing PC to send the halt and goto commands to the microcontroller. The microcontroller shall indicate the receipt of these Ethernet messages via a light or console message. The microcontroller shall also demonstrate the ability to differentiate the two messages.

Result: The team was able to pass this test using UDP streaming during January, but upon further investigation they found that TCP/IP streaming was a better choice and worked to pass the test again with that option. They were able to pass this test again in mid-April with the TCP/IP system. Team member Phil Fynan produced a test program named franken stub for this endeavor allowing team member Jake Roorda to write the needed Arduino code to read in the Ethernet buffer and extract the message fields.

**Actuation Testing Milestone 2:** Send Ethernet communications to the host computer

Test: The microcontroller on a development board shall be connected to its Ethernet hardware which will be connected to a network containing the test PC. An application of the tester’s choice shall be used to monitor Ethernet traffic sent to the testing PC. The microcontroller shall send two or more different
messages to the testing PC. The application of the testing PC shall be used to verify that the received messages match those that were programmed to be sent.

Result: The team was able to pass this test in mid-February using slightly modified example code built to send out HTTP messages. A standard web-browser and later the network debugging program Netcat were used to assess the messages being received.

**Actuation Testing Milestone 3: Sense Pan Position**

Test: The microcontroller on a development board shall be connected to the pan sensing hardware and a serial console of some form. The pan axis of the mirror pointing system shall be moved either manually or electrically from end to end with each complete motion taking at least 4 seconds. The tester shall visually ensure that the sensed data on the serial console matches the pan motion both in terms of position and speed.

Result: The team was able to pass this test in January. The final design of the system accomplishes angular sensing by using the built in motor controllers, and only limit switches are used to sense the motion of the entire system. Test code was used to read the motor encoders, keep track of how many turns the motor had made and print this value to a serial terminal. By rotating the motor shaft by hand and running the motor the team could verify that the counts were accurate and could keep up with the full speed of the motor. Actual returned values for tilt ranged linearly from 0x0000 to 0x2000 and for pan ranged linearly from 0x0000 to 0x0200.

**Actuation Testing Milestone 4: Sense Tilt Position**

Test: The microcontroller on a development board shall be connected to the tilt sensing hardware and a serial console of some form. The tilt axis of the mirror pointing system shall be moved either manually or electrically from end to end with each complete motion taking at least 4 seconds. The tester shall visually ensure that the sensed data on the serial console is consistent with the pan motion both in terms of position and speed. This can be done by marking positions on the motor and verifying that the encoder value at each position does not shift by more than 0.5% of full scale after 30 seconds of moving the motor and returning to the location.

Result: The team was also able to pass this test in January using a method very similar to that of actuation testing milestone 3. The only change was that a power supply was used to drive the motor as it could not be back-driven from the output shaft safely. Please see that test 3 for full details.

**Actuation Testing Milestone 5: Control Pan Motor**
Test: The microcontroller on a development board shall be connected to the pan motor controller which in turn shall be connected to the pan motor and the appropriate power supply. The microcontroller shall command the motor to rotate for one second in the clockwise direction followed by one second in the counterclockwise direction. The tester shall visually verify that the motor rotates as commanded each time. The time of each rotation phase shall be 1.0 seconds +/- 0.1 seconds.

Result: The team was able to pass this test in late December. The TI motor driver chip referenced earlier in this chapter was used to control the BLDC pan motor. Test code was used to command the requested rotation. Once the phase command table and encoders were synchronized the motor responded as commanded. Further testing beyond this milestone was also conducted including using a potentiometer to control the motor speed variably.

**Actuation Testing Milestone 6: Control Tilt Motor**

Test: The microcontroller on a development board shall be connected to the tilt motor controller which in turn shall be connected to the tilt motor and the appropriate power supply. The microcontroller shall command the motor to rotate for one second in the clockwise direction followed by one second in the counterclockwise direction. The tester shall visually verify that the motor rotates as commanded each time.

Result: The team was able to pass this test in early-February, shortly after receiving the tilt motor. The Freescale motor driver referenced earlier in this chapter was used to drive the motor in the required motions and the tester was able to verify that the motor responded as desired.

**Actuation Testing Milestone 7: Integrate sensors and motors for closed loop control**

Test: The microcontroller on a development board or in prototyped form shall be connected to a terminal and to both motor controllers which in turn shall be connected to their appropriate motors and power supplies. The microcontroller shall also be connected to the mirror and motor positioning sensors. The microcontroller shall recognize at least four different position commands representing 0%, 33%, 66% and 100% of the full scale range. The tester shall visually verify that the system moves and shall use a protractor to verify that the target positions were reached within 2°.

Result: The team was able to meet a modified version of this test in mid-February. Each motor was tested separately, as they are driven by separate controllers. A test program used the encoders to sense the motor position and used a PID loop to drive the motors as close as possible to that position. With some tuning both motors were able to hit their target values within 1°.
11 System Integration

The team has carefully made sure that the interface specifications between the different components are well defined so that each designer knows what to expect coming in and going out. The implementation details of these communications are defined in the subsystem chapters and the communication chapter. The system integration chapter discusses functionality that will be needed to assure successful integration and the necessary integration testing and debugging.

11.1 System Level Test

It is important to define a series of tests that a given system shall pass in the design phase. This prevents one from changing the requirements to suit what was produced at the expense of the product and the end user. These tests can be used as the device is built to help guide work to areas where it will have the most influence on meeting customer expectations. The final test will be to test the fully assembled system against the main functional goals listed in the tests below.

Test 1: Tracking software receives video feed from sensing unit

Test: The sensing system shall be connected via Ethernet to the tracking system. The tracking system shall visibly display the video feed from the sensing unit. The tester shall move their hand in front of the sensing camera and verify that this wave is visible on the tracking system screen within the required 0.3 seconds.

Results: This test was passed in mid-April.

Test 2: Tracking software can identify and track a given person on stage

Test: The sensing system shall be connected via Ethernet to the tracking system. The sensing system shall be placed near the back of an auditorium and pointed at the stage. A tester shall walk on stage. A second tester shall verify that the tracking system designates the location of the actor correctly within 2 feet as they move on stage.

Results: This test was often performed using prerecorded video segments. Video testing showed that the software was able to track the beacon within one foot.

Test 3: The control software shall be able to turn the light on and off and alter its intensity

Test: The control system shall be connected via Ethernet to the Actuation system. The lighting system shall be connected via SPI to the Actuation System. A tester shall use the ncat terminal program to
command the light to turn on and off repeatedly with various intensities. The tester shall visually verify that the light responds to these commands as expected.

**Results:** This test was passed 5/8/14. The light responded to all commands and did change intensity though less than expected by the team. This was deemed to be an issue with the commercial controller that the team could not directly address in the given time.

**Test 4:** Integrate Ethernet communication to control the mirror position

Test: This test shall be the same as Actuation Testing Milestone 8 with the exception that the commands shall be sent over Ethernet from the testing computer using an application of the testers choosing.

Result: This test was passed 5/6/14. The system performed identically to actuation testing milestone 8.

**Test 5:** The light shall be able to be manually pointed at any point on stage

Test: The setup shall be identical to Test 3 in an auditorium. A tester shall use the control system to manually point the light around the stage. The tester shall verify that the light can reach every portion of the stage including the wings if applicable and sweep through the stage in under 4 seconds.

Results: This test was passed 5/9/14. The light could move from corner to the opposite corner of its tracking range in 2 seconds.

**Test 6:** When tracking the light shall move and follow the desired person such that they remain entirely in the light

Test: The entire system shall be connected. The tester shall command the light to track a second tester standing on a stage. The second tester shall move around the stage at a fast walk. While they do this the first tester shall verify that they remain in the light of the tracking spotlight at all times.

Results: This test was passed on 5/12/14. Video of this test is linked from [http://www.calvin.edu/academic/engineering/2013-14-team9/UpdatesPage.html](http://www.calvin.edu/academic/engineering/2013-14-team9/UpdatesPage.html).

**Test 7:** The system presents minimal falling hazard

Test: Any part of the system that is mounted above the audience shall be tested for falling hazards by visually inspecting the system for any loose components or single clamping mechanisms holding suspended pieces. The system shall also be shaken vigorously by hand in all 3 axis. This test shall be considered passed if the system remains in one connected piece that could be supported entirely by the safety chain. This test will be considered a failure in any piece of the system hits the ground or if the safety chain is visibly damaged by the system.
Results: This test has been performed numerous times as the system is built up and any failures have been immediately rectified.

**Test 10:** The system functions with minimal safety hazards to people in the vicinity

Test: Self-audits and independent audits conducted by other teams will be used to access the safety of the system throughout the build and test process. Auditors shall be specifically mindful of hazards related to heights, electricity, fire, bright lighting, trip hazards, privacy, and the possibility of being caught in a moving mechanism.

Results: These audits have been conducted regularly and all issues were addressed as soon as they were recognized.

### 11.2 System Level Debug

When a test fails, it is important to quickly isolate and fix the problem. To aid in this, whenever feasible, each subsystem shall be able to be integrated independently so faults can be identified one system at a time. It is also important that each subsystem is thoroughly tested and peer reviewed before integration begins. Finding problems prior to integration will allow for much faster fixes than once integration has started. The subsystems shall also be built with the final test and debug in mind. Whenever possible the team will build in diagnostic data outputs so that the system can be debugged more easily.

When a problem is found or a requirement is not met, the team shall begin by qualifying the failure state to gain an understanding of what occurred. In software, this would involve looking at variable values and code execution paths while in hardware this could include looking and logging voltage levels with an oscilloscope or recording the system motion with a camera. The error shall also be reproduced if it is possible and safe to do so.

Once the problem state is well defined, the team will look into what the most likely root cause of the failure state was. Once the root cause is identified, the team will brainstorm and propose to the group solutions to fix the problem. The team will pick a solution based upon the time and monetary cost of the solutions and their effectiveness in addressing the problem condition. If no root cause could be identified, the team shall formulate and perform tests to identify the root cause.

During the second semester this method was followed well. The team was able to troubleshoot many issues related to Ethernet communication, LED drivers, and tracking. Many of the boards were built incrementally to allow for step by step debugging. The team also wrote a number of utility and testing programs to assess aspects of the project individually.
12 Business Plan

Calvin College desires that its engineering graduates be able to function both in a corporate environment as well as an academic environment. For this reason, one of the requirements of the Senior Design course is for each team to formulate a potential business venture based on their senior project design. This chapter will look into the management and financials of a company based on selling the autonomous tracking spotlight system.

12.1 Vision

Spotlights are used in a variety of stage productions ranging from musicals and plays to concerts and business events. They are a staple in modern lighting for their ability to keep one subject in bright light while keeping the rest of the stage in blackout. However, they present a great deal of cost to event producers. Each spotlight requires its own operator, so there are additional costs associated with hiring these operators. The spotlights must also be coordinated over an intercom system and not electrically as would be preferred.

The team’s robotic spotlight system will have many advantages over a traditional spotlight system. This system will require only one operator to control multiple spotlights via a simple graphical control interface. This operator could also run other lighting controls in the control booth. This would be a significant savings in labor costs to the producers. This spotlight will autonomously track its intended subject in order to free the operator for other tasks. The team also designed the spotlight to be highly energy efficient.

The team was careful to consider the ethical implications of their device. While the spotlight system will present many benefits to venue owners, one of the primary benefits is the reduced operator cost. This cost is reduced through the elimination of workers. This brings up many issues related with justice and caring. As Christians the team wants to treat all people fairly, and not create a system that benefits wealthy theatre owners while hurting working class spotlight operators.

To better understand the issues, the team looked into typical spotlight operators. While large concerts may have dedicated spotlight operators, in most concerts the spotlight operator will double as road crew and their primary responsibility is to move equipment. Spotlight operators also face many hazards including burns from hot spotlights and the height of their working platform. They also can suffer hearing loss from loud music.

From team members personal experience they also found that in general most productions base their lighting and effects on a given budget, so cost savings in one area may go back into another area of the
performance. As is the case with most industrial automation there will be a direct loss of jobs that are replaced, but more jobs will be created in support industries. Our hope is that existing spotlight operators can learn how to install, operate, and maintain the system.  

12.2 Sales Volume Estimates

The production cost estimate is based on a total annual sales volume of 2500 units. This number was based on using Grand Rapids as a representative sample for the United States. In the greater Grand Rapids area, the team estimated that there were approximately 40 venues that would use spotlights based on a Google Maps search. These include theatre groups, high schools and colleges, and concert venues. The average number of spotlights per venue was estimated to be three. Most groups will use two spotlights, but some will use many more. Thus the size of the Grand Rapids market is 120 spotlights.

The population of the greater Grand Rapids area is approximately 1 million people. The US has a population of around 317 million people. Assuming Grand Rapids to be a representative sample, this places the overall spotlight market at about 38000 spotlights. Assuming each spotlight has a useful life of 5 years, this places the annual spotlight market at 7600 spotlights. It is believed that this new product could capture 1/3rd of this market. This places annual sales around 2500 units.

There are several risky assumptions in the analysis above. It seems unlikely that the team would be able to obtain 33% of the market in one year and continue this trend from one year to another. Nevertheless, with additional certifications and a few power supply modifications the team could sell the system outside of US which would increase sales potential. Moreover, large touring concerts were left out of the analysis, and they could be large potential customer bases. Hence, when these factors are combined, they add some unknowns to the sales estimate. However, they do not change it considerably.

12.3 Fixed Production Costs

The fixed costs for the system are mainly the engineering design time and startup costs of manufacturing and production. There are also other fixed costs associated with management, accounting, marketing, and customer service staff time. Since this would be a small company, the management and accounting hours were estimated to be 20% of the engineering hours.

Marketing costs assumed attending the LDI lighting trade show in Las Vegas with a 30 foot by 30 foot booth space. This total came to $23,850 for the floor space and another $6,000 for transportation and staffing costs. In addition to this, the company would also place half page ads in each issue of a stage lighting journal like Projection, Lights, and Staging News. This cost came to about $9,000 for ads in two
journals. An additional $10,000 would be allocated for web video development. Thus the total marketing budget was set at $60,000.\textsuperscript{141} These fixed costs are shown in Table 14 below and assume $100 per hour labor costs.

### Table 14: Fixed Costs

<table>
<thead>
<tr>
<th>Components</th>
<th>Part(s)</th>
<th>Cost</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor Unit</td>
<td>Case Start-Up</td>
<td>$50,000</td>
<td>Plastic Injection Molding Startup Cost</td>
</tr>
<tr>
<td>Control system</td>
<td>Case</td>
<td>$75,000</td>
<td>Plastic Injection Molding Startup Cost</td>
</tr>
<tr>
<td>Actuation</td>
<td>Bracket tooling</td>
<td>$30,000</td>
<td>Sheet metal assembly</td>
</tr>
<tr>
<td>Lighting</td>
<td>Case</td>
<td>$50,000</td>
<td>Metal Tooling Costs</td>
</tr>
<tr>
<td>Marketing</td>
<td></td>
<td>$60,000</td>
<td></td>
</tr>
<tr>
<td>Rent and Utilities</td>
<td></td>
<td>$78,000</td>
<td>Small Office and Assembly Area</td>
</tr>
<tr>
<td><strong>Subtotal</strong></td>
<td></td>
<td>$461,000</td>
<td></td>
</tr>
</tbody>
</table>

### Fixed Labor

<table>
<thead>
<tr>
<th>Task</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prototyping</td>
<td>350</td>
</tr>
<tr>
<td>Sensor Unit Design</td>
<td>200</td>
</tr>
<tr>
<td>Control Software</td>
<td>300</td>
</tr>
<tr>
<td>Hardware Design</td>
<td>150</td>
</tr>
<tr>
<td>Lighting Electronic</td>
<td>30</td>
</tr>
<tr>
<td>Marketing</td>
<td>50</td>
</tr>
<tr>
<td>Test and Integration</td>
<td>200</td>
</tr>
</tbody>
</table>
12.4 Variable Production Costs

The variable costs for our product are mainly component part costs and assembly hours. The total assembly time for our product is estimated to be 1.5 hours with an additional hour of labor related to the product volume. This is low given the complexity of the overall system because the team will use part vendors to do a lot of assembly for them including PCB population and test. The detailed variable costs are shown in Table 15 Below.

Assembly costs were based on the estimated time needed to assemble the components of each system. This including mounting circuit boards, plugging in connectors, assembling the case, testing the unit, fixing any issues, packaging, and shipment. Additionally it was assumed that it would take a half-hour of sales time to sell each unit, and that each unit would generate on average 12 min of warranty repair work in the future.

Component costs were estimated using data from high volume electronics vendors assuming a single year of part demand. The team specifically looked at similar products from DigiKey, Mouser, TTI, and Avnet. Other specialty parts like optics and cases were estimated using an assumed 50% discount on the individual piece prices from the prototype system.

<table>
<thead>
<tr>
<th>Table 15: Variable Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Subsystem</strong></td>
</tr>
<tr>
<td>Sensor Unit</td>
</tr>
<tr>
<td>IR Camera</td>
</tr>
<tr>
<td>Visual Camera</td>
</tr>
<tr>
<td>Other logic (Microcontroller)</td>
</tr>
<tr>
<td>Component</td>
</tr>
<tr>
<td>----------------------------------</td>
</tr>
<tr>
<td>Ethernet driver</td>
</tr>
<tr>
<td>Case (plastic)</td>
</tr>
<tr>
<td>PCB</td>
</tr>
<tr>
<td><strong>Subtotal</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control system</td>
<td></td>
</tr>
<tr>
<td>Assembly Labor</td>
<td>0.25</td>
</tr>
<tr>
<td>Microcontroller</td>
<td>$3.00</td>
</tr>
<tr>
<td>USB</td>
<td>$1.00</td>
</tr>
<tr>
<td>Case (plastic)</td>
<td>$2.00</td>
</tr>
<tr>
<td>DMX Communication</td>
<td>$3.00</td>
</tr>
<tr>
<td>PCB</td>
<td>$0.84</td>
</tr>
<tr>
<td><strong>Subtotal</strong></td>
<td><strong>$9.84</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actuation</td>
<td></td>
</tr>
<tr>
<td>Assembly Labor</td>
<td>0.5</td>
</tr>
<tr>
<td>Motors with gears (x2)</td>
<td>$35.00</td>
</tr>
<tr>
<td>Brackets</td>
<td>$8.00</td>
</tr>
<tr>
<td>Microcontroller</td>
<td>$0.45</td>
</tr>
<tr>
<td>Motor control</td>
<td>$1.00</td>
</tr>
<tr>
<td>Connector</td>
<td>$2.00</td>
</tr>
<tr>
<td>Mirror</td>
<td>$20.00</td>
</tr>
<tr>
<td>Power Supply</td>
<td>$0.50</td>
</tr>
<tr>
<td><strong>Subtotal</strong></td>
<td><strong>$66.95</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lighting</td>
<td></td>
</tr>
<tr>
<td>Assembly Labor</td>
<td>0.5</td>
</tr>
<tr>
<td>Material</td>
<td>Cost</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>LED light</td>
<td>$20.00</td>
</tr>
<tr>
<td>Driver</td>
<td>$1.00</td>
</tr>
<tr>
<td>Microcontroller</td>
<td>$1.00</td>
</tr>
<tr>
<td>Connector</td>
<td>$1.00</td>
</tr>
<tr>
<td>Light Frame</td>
<td>$12.00</td>
</tr>
<tr>
<td>Optics</td>
<td>$15.00</td>
</tr>
<tr>
<td><strong>Subtotal</strong></td>
<td><strong>$50.00</strong></td>
</tr>
</tbody>
</table>

**Other Labor**

<table>
<thead>
<tr>
<th>Activity</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testing</td>
<td>0.3</td>
</tr>
<tr>
<td>Warranty repair</td>
<td>0.2</td>
</tr>
<tr>
<td>Sales</td>
<td>0.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Material Cost per Unit</th>
<th>$144.63</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost of Assembly</td>
<td>$250</td>
</tr>
<tr>
<td>2.5 Hours</td>
<td></td>
</tr>
<tr>
<td>Variable Unit Cost</td>
<td>$394.63</td>
</tr>
</tbody>
</table>

### 12.5 Profitability

Based on the information given above, the annual cost of producing 2500 units is approximately 1.5 million dollars. This will break down to a cost of $638.50 per unit. The team believes that the market will support a selling price between $1500 and $6000 for the entire system. The team estimates the retail margin in this sector to be roughly 100% based on personal experience with the similar musical instrument market. This gives a sales revenue per unit between $750 and $3000. Subtracting the cost of the system gives a net profit between $110 and $2360 per unit.

Existing spotlights with this level of light output sell for between $800 and $1200, but with the tracking technology these lights become significantly more valuable. The closest competitor to this system was the Wybron Autopilot II, which is no longer on the market. It retailed for $33,850, but the team believes
that this high price is partially what led to the failure and discontinuation of this system.\textsuperscript{143} The team would also consider selling individual components of our system, so that a large concert can control many lights with one controlling computer, but that goes beyond the scope of this analysis.
13 Conclusion

The team believes that the project has met constraints design requirements, cost, and schedule while successfully managing risk. In terms of schedule, the total estimated work for the project was 740 hours, which works out to around 10 hours per week per team member. These weekly hour levels have been handled in the first semester, so it was assumed it would likely be manageable continuing into next semester. The time spent on the project was approximately 1150 hours which means weekly hour estimates where off by about 65%, but this was manageable. Also, the critical path identified in the WBS flowchart had the prototype constructed around three weeks before the final demonstration date in May. The actual completion time was about 1 week before the demonstration date which shows that the flow chart planning successfully drove the project to complete before the deadline, but that important deadlines were allowed to slip. The degree of error in completion time was planned ahead of time in case of extra time required as was actually experienced. In terms of cost, the approved budget for the project was $775 which is near the estimated course average of $500 per team. The actual cost of the system was $780.77 which is just 0.8% over budget. Lastly, in terms of technology, a tracking prototype has been demonstrated that can track an object using a webcam and a fabricated IR beacon with the video of the demonstration available in the team website. Hardware testing for the lighting subsystem and actuation have proved successful. The actuation testing shows the pulley and motor configurations operating correctly. Lighting testing shows stable light output with user control. Complete system testing was quite successful, showing that the light can track a user in darkness effectively. For a video of this working see http://www.calvin.edu/academic/engineering/2013-14-team9/index.html.

In the future this project could be further improved in a number of ways. More capable video hardware could certainly aid in tracking. Specifically higher frame rates would allow the tracking software to make better use of the specific beacon flashing pattern. The mirror pointing system could also be made lighter and more rigid to eliminate much of the shaking present in the current design. A larger budget would also permit the use of more standard optics which could allow the spotlight to have a more focused beam and a larger throw distance. A microcontroller with more GPIO pins would also be desirable for the pan subsystem to allow for future expansion if necessary. Lastly, it would be an interesting expansion to make a flashing IR laser pointer that could be used by a lighting tech to manually point the light from anywhere in the theatre.
14 Acknowledgements

14.1 Calvin Community

Team 9 would like to thank team advisor Professor VanderLeest for his guidance and motivation as we completed this project and for his 298 comments on the PPFS draft and 308 comments on the Final Report draft.

Team 9 also thanks Bob DeKraker, Phil Jasperse, and Chuck Holwerda for their practical help in relation to prototype production and providing the team with lab equipment.

Team 9 would also like to thank the Calvin College Office of Campus Events and Calvin Theatre Company for taking the time to discuss the functionality and technology of spotlighting systems used in Calvin’s Theater venues.

Team 9 also thanks the reviewers Eric Walstra, Paul Rozeboom, and Mark Michmerhuizen on the CEAC who took the time to review the project and provide project feedback after the review.

14.1 Others

Team 9 would like to thank Tim Theriault of GE Aviation Systems for offering advice in terms of project management.

The members of team 9 would like to thank their families, friends, and roommates who have been supportive in this endeavor and who daily inspired them to do their best.

14.2 Industrial

The team would like to express its thanks to Texas Instruments for providing samples of the TLC5916 8-bit constant current sink and the DRV8313PWPR triple-half-bridge driver for this project free of charge.
15 References


11. Conversation with Steve Haase of Calvin College Theater Company


18. Conversation with Doug Huizinga of the Calvin College Office of Campus Events


28 Photo Credit: http://media.digikey.com/Photos/Bridgelux/MFG_Vero%2029.jpg
35 IBID
58 Photo Credit: http://www.hitechnic.com/PDGImages/BeaconAx200.jpg


Code accessible from https://github.com/pfynan/spoor/blob/master/shoggoth.thrift


IBID


