UAV Landingpad

Senior Design Final Documentation

Team Name

Steven Huerta Christopher Smith Dylan Geyer Jonathan Dixon

November 4, 2015

Contents

List of Figures

List of Tables

List of Algorithms

Overview Statements

0.1 Mission Statement

Mission statement inserted here.

0.2 Elevator Pitch

Elevator Pitch inserted here.

Document Preparation and Updates

Current Version [X.X.X]

Prepared By:
Team Member #1
Team Member #2
Team Member #3

Revision History

Date	***Author***	***Version***	***Comments***
2/2/12	*Team Member #1*	*1.0.0*	*Initial version*
3/4/12	*Team Member #3*	*1.1.0*	*Edited version*

1

Overview and concept of operations

The overview should take the form of an executive summary. Give the reader a feel for the purpose of the document, what is contained in the document, and an idea of the purpose for the system or product.

1.1 Scope

What scope does this document cover?

1.2 Purpose

What is the purpose of the system or product?

1.2.1 Major System Component #1

Describe briefly the role this major component plays in this system.

1.2.2 Major System Component #2

Describe briefly the role this major component plays in this system.

1.2.3 Major System Component #3

Describe briefly the role this major component plays in this system.

1.3 Systems Goals

Briefly describe the overall goals this system plans to achieve. These goals are typically provided by the stakeholders. This is not intended to be a detailed requirements listing. Keep in mind that this section is still part of the Overview.

1.4 System Overview and Diagram

Provide a more detailed description of the major system components without getting too detailed. This section should contain a high-level block and/or flow diagram of the system highlighting the major components. See Figure 1.1. This is a floating figure environment. LaTeX will try to put it close to where it was typeset but will not allow the figure to be split if moving it can not happen. Figures, tables, algorithms and many other floating environments are automatically numbered and placed in the appropriate type of table of contents. You can move these and the numbers will update correctly.

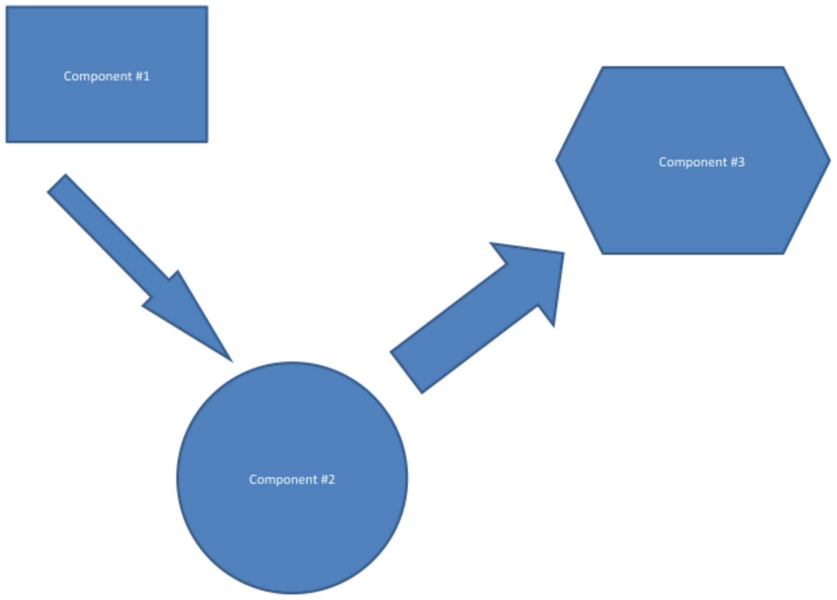

Figure 1.1: A sample figure .... System Diagram

1.5 Technologies Overview

This section should contain a list of specific technologies used to develop the system. The list should contain the name of the technology, brief description, link to reference material for further understanding, and briefly how/where/why it was used in the system. See Table 1.1. This is a floating table environment. LaTeX will try to put it close to where it was typeset but will not allow the table to be split.

Table 1.1: A sample Table ... some numbers.

7C0	hexadecimal
3700	octal
11111000000	binary
1984	decimal

2 Project Overview

This section provides some housekeeping type of information with regard to the team, project, etc.

2.1 Team Members and Roles

Describe who was involved and what role(s) were played.

2.2 Project Management Approach

This section will provide an explanation of the basic approach to managing the project. Typically, this would detail how the project will be managed through a given Agile methodology. The sprint length (i.e. 2 weeks) and product backlog ownership and location (ex. Trello) are examples of what will be discussed. An overview of the system used to track sprint tasks, bug or trouble tickets, and user stories would be warranted.

2.3 Phase Overview

If the system will be implemented in phases, describe those phases/sub-phases (design, implementation, testing, delivery) and the various milestones in this section. This section should also contain a correlation between the phases of development and the associated versioning of the system, i.e. major version, minor version, revision.

2.4 Terminology and Acronyms

Provide a list of terms used in the document that warrant definition. Consider industry or domain specific terms and acronyms as well as system specific.

3

Design and Implementation

This section is used to describe the design details for each of the major components in the system. Note that this chapter is critical for all tracks. Research tracks would do experimental design here where other tracks would include the engineering design aspects. This section is not brief and requires the necessary detail that can be used by the reader to truly understand the architecture and implementation details without having to dig into the code. Sample algorithm: Algorithm 1. This algorithm environment is automatically placed - meaning it floats. You don't have to worry about placement or numbering.

Algorithm 1 Calculate $y = x^n$

```
Require: $n \geq 0 \vee x \neq 0$
Ensure: $y = x^n$
  $y \Leftarrow 1$
  if $n < 0$ then
    $X \Leftarrow 1/x$
    $N \Leftarrow -n$
  else
    $X \Leftarrow x$
    $N \Leftarrow n$
  end if
  while $N \neq 0$ do
    if $N$ is even then
      $X \Leftarrow X \times X$
      $N \Leftarrow N/2$
    else {$N$ is odd}
      $y \Leftarrow y \times X$
      $N \Leftarrow N - 1$
    end if
  end while
```

Citations look like [**?**, **?**, **?**] and [**?**, **?**, **?**]. These are done automatically. Just fill in the database `designrefs.bib` using the same field structure as the other entries. Then pdflatex the document, bibtex the document and pdflatex twice again. The first pdflatex creates requests for bibliography entries. The bibtex extracts and formats the requested entries. The next pdflatex puts them in order and assigns labels. The final pdflatex replaces references in the text with the assigned labels. The bibliography is automatically constructed.

3.1 Major Component #1

3.1.1 Technologies Used

This section provides a list of technologies used for this component. The details for the technologies have already been provided in the Overview section.

3.1.2 Component Overview

This section can take the form of a list of features.

3.1.3 Phase Overview

This is an extension of the Phase Overview above, but specific to this component. It is meant to be basically a brief list with space for marking the phase status.

3.1.4 Architecture Diagram

It is important to build and maintain an architecture diagram. However, it may be that a component is best described visually with a data flow diagram.

3.1.5 Data Flow Diagram

It is important to build and maintain a data flow diagram. However, it may be that a component is best described visually with an architecture diagram.

3.1.6 Design Details

This is where the details are presented and may contain subsections. Here is an example code listing:

```c
#include <stdio.h>
#define N 10
/* Block
 * comment */

int main()
{
    int i;

    // Line comment.
    puts("Hello world!");

    for (i = 0; i < N; i++)
    {
        puts("LaTeX is also great for programmers!");
    }

    return 0;
}
```

This code listing is not floating or automatically numbered. If you want auto-numbering, but it in the algorithm environment (not algorithmic however) shown above.

3.2 Major Component #2

3.2.1 Technologies Used

This section provides a list of technologies used for this component. The details for the technologies have already been provided in the Overview section.

3.2.2 Component Overview

This section can take the form of a list of features.

3.2.3 Phase Overview

This is an extension of the Phase Overview above, but specific to this component. It is meant to be basically a brief list with space for marking the phase status.

3.2.4 Architecture Diagram

It is important to build and maintain an architecture diagram. However, it may be that a component is best described visually with a data flow diagram.

3.2.5 Data Flow Diagram

It is important to build and maintain a data flow diagram. However, it may be that a component is best described visually with an architecture diagram.

3.2.6 Design Details

This is where the details are presented and may contain subsections.

3.3 Major Component #3

3.3.1 Technologies Used

This section provides a list of technologies used for this component. The details for the technologies have already been provided in the Overview section.

3.3.2 Component Overview

This section can take the form of a list of features.

3.3.3 Phase Overview

This is an extension of the Phase Overview above, but specific to this component. It is meant to be basically a brief list with space for marking the phase status.

3.3.4 Architecture Diagram

It is important to build and maintain an architecture diagram. However, it may be that a component is best described visually with a data flow diagram.

3.3.5 Data Flow Diagram

It is important to build and maintain a data flow diagram. However, it may be that a component is best described visually with an architecture diagram.

3.3.6 Design Details

This is where the details are presented and may contain subsections.

4

System and Unit Testing

This section describes the approach taken with regard to system and unit testing.

4.1 Overview

Provides a brief overview of the testing approach, testing frameworks, and general how testing is/will be done to provide a measure of success for the system.

4.2 Dependencies

Describe the basic dependencies which should include unit testing frameworks and reference material.

4.3 Test Setup and Execution

Describe how test cases were developed, setup, and executed. This section can be extremely involved if a complete list of test cases was warranted for the system.

5 Development Environment

The basic purpose for this section is to give a developer all of the necessary information to setup their development environment to run, test, and/or develop.

5.1 Development IDE and Tools

Describe which IDE and provide links to installs and/or reference material.

5.2 Source Control

Which source control system is/was used? How was it setup? How does a developer connect to it?

5.3 Dependencies

Describe all dependencies associated with developing the system.

5.4 Build Environment

How are the packages built? Are there build scripts?

5.5 Development Machine Setup

If warranted, provide a list of steps and details associated with setting up a machine for use by a developer.

6

User Documentation

This section should contain the basis for any end user documentation for the system. End user documentation would cover the basic steps for setup and use of the system. It is likely that the majority of this section would be present in its own document to be delivered to the end user. However, it is recommended the original is contained and maintained in this document.

6.1 User Guide

The source for the user guide can go here. You have some options for how to handle the user docs. If you have some `newpage` commands around the guide then you can just print out those pages. If a different formatting is required, then have the source in a separate file `userguide.tex` and include that file here. That file can also be included into a driver (like the senior design template) which has the client specified formatting. Again, this is a single source approach.

6.2 Installation Guide

6.2.1 Environment Setup

The Environment we will be using is Ubuntu 14.04 Long Term Support(LTS), and Robotics Operating System (ROS). The distribution of ROS will be Jade which was released in May 2015.

6.3 Programmer Manual

7 Class Index

7.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

8

Class Documentation

8.1 Poly Class Reference

Public Member Functions

- Poly ()
- ~Poly ()
- int myfunction (int)

8.1.1 Constructor & Destructor Documentation

8.1.1.a Poly::Poly ( )

My constructor

8.1.1.b Poly::~Poly ( )

My destructor

8.1.2 Member Function Documentation

8.1.2.a int Poly::myfunction (int *a*)

my own example function fancy new function

new variable

The documentation for this class was generated from the following file:

- hello.cpp

9

Experimental Logs

For research projects one needs to keep a log of all research/lab activities.

9.1 Takeoff Log

10/15/15 Ran modified filter on data sets 1 - 6. Results were ... First Last

9.2 Navigation Log

10/03/15 Tested ardupilot with windows. Could not get landingpad ardupilot to connect, but was able to get Robotics Team ardupilot to work with windows. Also was able to get this ardupilot connected in Ubuntu and working with MavRos. Chris Smith

9.3 Landing Log

10/15/15 Ran modified filter on data sets 1 - 6. Results were ... First Last

9.4 Mechanical Log

9.4.1 UAV

10/15/15 Ran modified filter on data sets 1 - 6. Results were ... First Last

9.4.2 UGV

10/15/15 Ran modified filter on data sets 1 - 6. Results were ... First Last

9.5 Simulation Log

9/26/15 Installed Ubuntu LTS, ROS Jade, and Gazebo5 Chris Smith

10/3/15 Went through tutorials on using Gazebo and Ros Jade. Chris Smith

10/9/15 Installed Rotors-Simulator for ROS and Gazebo. Chris Smith

10/15/15 Worked on Gazebo Simulation with Rotors-Simulator. Chris Smith

10

Research Results

This chapter describes the results and conclusions of your research. This would be the final report for a research project.

10.1 Result 1

10.2 Result 2

10.3 Conclusions

10.4 Further work

Bibliography

SDSMT SENIOR DESIGN
SOFTWARE DEVELOPMENT AGREEMENT

This Software Development Agreement (the “Agreement”) is made between the SDSMT Computer Science

Senior Design Team ____,
(“Student Group”)

consisting of team members ____,
(“Student Names”)

and Sponsor ____,
(“Company Name”)

with address: ____.

[Note: Bracketed material is included to suggest content that will vary with each agreement. I STRONGLY SUGGEST THAT THE INSTRUCTOR LOOK AT THE COMPLETED AGREEMENT BEFORE YOU SIGN IT!!]

1 RECITALS

1. Sponsor desires Senior Design Team to develop software [for use in Sponsor’s simulation platform for optical fiber transmissions of digitized video signals] (the ”Field”).
2. Senior Design Teams willing to develop such Software.

NOW, THEREFORE, in consideration of the mutual covenants and promises herein contained, the Team and Sponsor agree as follows:

2 EFFECTIVE DATE

This Agreement shall be effective as of ____________________ (the “Effective Date”).

3 DEFINITIONS

1. “Software” shall mean [the computer programs in machine readable object code form and any subsequent error corrections or updates supplied to Sponsor by Senior Design Team pursuant to this Agreement.] [Depending on the particulars of each agreement, any or all of the following may need to be specified. If they are relevant, they should be used throughout, modifying the standard form as appropriate.]
2. “Acceptance Criteria” means the written technical and operational performance and functional criteria and documentation standards set out in the [project plan.]
3. “Acceptance Date” means [the date for each Milestone when all Deliverables included in that Milestone have been accepted by Sponsor in accordance with the Acceptance Criteria and this Agreement.]
4. “Deliverable” means a deliverable specified in the [project plan.]
5. “Delivery Date” shall mean, [with respect to a particular Milestone,] the date on which University has delivered to Sponsor all of the Deliverables [for that Milestone] in accordance with [the project plan and] this Agreement.

6. "Documentation" means the documents, manuals and written materials (including end-user manuals) referenced, indicated or described in [the project plan] or otherwise developed pursuant to this Agreement.
7. "Milestone" means the completion and delivery of all of the Deliverables or other events which are included or described in [the project plan] scheduled for delivery and/or completion on a given target date; a Milestone will not be considered completed until the Acceptance Date has occurred with respect to all of the Deliverables for that Milestone.

4 DEVELOPMENT OF SOFTWARE

1. Senior Design Team will use its best efforts to develop the Software described in [the project plan.] The Software development will be under the direction of or his/her successors as mutually agreed to by the parties ("Team Lead") and will be conducted by the Team Lead. The Team will deliver the Software to the satisfaction of the course instructor that reasonable effort has been made to address the needs of the client. The Team understands that failure to deliver the Software is grounds for failing the course.
2. Sponsor understands that the Senior Design course's mission is education and advancement of knowledge, and, consequently, the development of Software must further that mission. The Senior Design Course does not guarantee specific results or any results, and the Software will be developed only on a best efforts basis. The Software is considered PROOF OF CONCEPT only and is NOT intended for commercial, medical, mission critical or industrial applications.
3. The Senior Design instructor will act as mediator between Sponsor and Team; and resolve any conflicts that may arise.

5 COMPENSATION

[This is entirely subject to negotiation. Normally NO COMPENSATION occurs in a Senior Design Project. On occasion an intern status and wage is appropriate.]

6 CONSULTATION AND REPORTS

1. Sponsor's designated representative for consultation and communications with the Team Lead shall be

   ______________________________ or such other person as Sponsor may from time to time designate to the Team Lead ("Designated Representative").
2. During the Term of the Agreement, Sponsor's representatives may consult informally with course instructor regarding the project, both personally and by telephone. Access to work carried on in University facilities, if any, in the course of this Agreement shall be entirely under the control of University personnel but shall be made available on a reasonable basis.
3. The Team Lead will submit written progress reports. At the conclusion of this Agreement, the Team Lead shall submit a comprehensive final report in the form of the formal course documentation at the conclusion of the Senior Design II course.

7 CONFIDENTIAL INFORMATION

1. The parties may wish, from time to time, in connection with work contemplated under this Agreement, to disclose confidential information to each other ("Confidential Information"). Each party will use reasonable efforts to prevent the disclosure of any of the other party's Confidential Information to third parties for

a period of three (3) years after the termination of this Agreement, provided that the recipient party's obligation shall not apply to information that:

   (a) is not disclosed in writing or reduced to writing and so marked with an appropriate confidentiality legend within thirty (30) days of disclosure;
   (b) is already in the recipient party's possession at the time of disclosure thereof;
   (c) is or later becomes part of the public domain through no fault of the recipient party;
   (d) is received from a third party having no obligations of confidentiality to the disclosing party;
   (e) is independently developed by the recipient party; or
   (f) is required by law or regulation to be disclosed.

2. In the event that information is required to be disclosed pursuant to subsection (6), the party required to make disclosure shall notify the other to allow that party to assert whatever exclusions or exemptions may be available to it under such law or regulation.

8 INTELLECTUAL PROPERTY RIGHTS

[Negotiated on a case-by-case basis. This must address who owns the algorithms and who owns the source code. For example: All deliverables become property of the Sponsor. Roughly: If the idea originates with the sponsor, or if a sponsor pays you to develop an idea, then they have legitimate claim to the IP. If the idea originates from the University (through faculty or staff) then the University has legitimate claim. If the idea is yours (student) and you develop it without external compensation then you have legitimate claim.]

9 WARRANTIES

The Senior Design Team represents and warrants to Sponsor that:

1. the Software is the original work of the Senior Design Team in each and all aspects;
2. the Software and its use do not infringe any copyright or trade secret rights of any third party.

No agreements will be made beyond items (1) and (2).

10 INDEMNITY

1. Sponsor is responsible for claims and damages, losses or expenses held against the Sponsor. [Sponsor may have something to add here.]
2. Sponsor shall indemnify and hold harmless the Senior Design Team, its affiliated companies and the officers, agents, directors and employees of the same from any and all claims and damages, losses or expenses, including attorney's fees, caused by any negligent act of Sponsor or any of Sponsor's agents, employees, subcontractors, or suppliers.
3. NEITHER PARTY TO THIS AGREEMENT NOR THEIR AFFILIATED COMPANIES, NOR THE OFFICERS, AGENTS, STUDENTS AND EMPLOYEES OF ANY OF THE FOREGOING, SHALL BE LIABLE TO ANY OTHER PARTY HERETO IN ANY ACTION OR CLAIM FOR CONSEQUENTIAL OR SPECIAL DAMAGES, LOSS OF PROFITS, LOSS OF OPPORTUNITY, LOSS OF PRODUCT OR LOSS OF USE, WHETHER THE ACTION IN WHICH RECOVERY OF DAMAGES IS SOUGHT IS BASED ON CONTRACT TORT (INCLUDING SOLE, CONCURRENT OR OTHER NEGLIGENCE AND STRICT

LIABILITY), STATUTE OR OTHERWISE. TO THE EXTENT PERMITTED BY LAW, ANY STATUTORY REMEDIES WHICH ARE INCONSISTENT WITH THE PROVISIONS OF THESE TERMS ARE WAIVED.

11 INDEPENDENT CONTRACTOR

For the purposes of this Agreement and all services to be provided hereunder, the parties shall be, and shall be deemed to be, independent contractors and not agents or employees of the other party. Neither party shall have authority to make any statements, representations or commitments of any kind, or to take any action which shall be binding on the other party, except as may be expressly provided for herein or authorized in writing.

12 TERM AND TERMINATION

1. This Agreement shall commence on the Effective Date and extend until the end of classes of the second semester of Senior Design (CSC 467), unless sooner terminated in accordance with the provisions of this Section ("Term").
2. This Agreement may be terminated by the written agreement of both parties.
3. In the event that either party shall be in default of its materials obligations under this Agreement and shall fail to remedy such default within thirty (30) days after receipt of written notice thereof, this Agreement shall terminate upon expiration of the thirty (30) day period.
4. Any provisions of this Agreement which by their nature extend beyond termination shall survive such termination.

13 ATTACHMENTS

Attachments A and B are incorporated and made a part of this Agreement for all purposes.

14 GENERAL

1. This Agreement constitutes the entire and only agreement between the parties relating to the Senior Design Course, and all prior negotiations, representations, agreements and understandings are superseded hereby. No agreements altering or supplementing the terms hereof may be made except by means of a written document signed by the duly authorized representatives of the parties.
2. This Agreement shall be governed by, construed, and enforced in accordance with the internal laws of the State of South Dakota.

15 SIGNATURES

Replace with name of student #1	Date
Replace with name of student #2	Date
Replace with name of student #3	Date
Replace with name of sponsor's representative	Date

A

Product Description

Write a description of the product to be developed. Use sectioning commands as neccessary.

NOTE: *This is part of the contract.*

B

Publications

Research Track: This chapter will include any publications generated from the research. Most likely these will be preprints and one will just include the pdf.

C

Sprint Reports

1 Sprint Report #1

Team Overview

Name
Expeditus

Members
Jonathan Dixon, Dylan Geyer, Steven Huerta, Christopher Smith

Project Title
UAV Landing Pad

Sponsor
Dr. Larry Pyeatt, SDSMT MCS

Sponsor Overview

Sponsor Description
The Math and Computer Science Department of South Dakota School of Mines and Technology, in addition to providing ABET certified education to students, conducts software-side robotics research including autonomy, navigation, and computer vision.

Sponsor Problem
The capability of UAVs to rapidly search a large area, especially one that is difficult to traverse by foot or vehicle, would be invaluable to operations such as search & rescue. However, small UAVs have a very limited flight time. A system incorporating a UVG equipped with a landing pad that also serves as a charging station would allow the UAV to be delivered to areas of limited access. The UAV could then, being provided with waypoints by the user, autonomously take-off, and navigate through the waypoints. After moving through the waypoints, or when the UAV requires recharging, the UAV will return to the UVG and safely land in such a way that the charging unit can connect to the UAV.

Sponsor Needs

- Ability to communicate waypoints to UAV.
- UAV can autonomously take-off.
- UAV can autonomously navigate through waypoints.
- UAV can autonomously navigate back to landing pad.
- UAV can autonomously land safely and with the correct orientation.

Project Overview

Phase 1 First phase will focus on finalizing the autonomous take-off and waypoint navigation by the UAV. Previous development will be reviewed, implemented, and tested. Simulation environment will be created for the purpose of testing landing algorithms.

Phase 2 Second phase will focus on finalizing autonomous landing

Project Environment

Project Boundaries

- Project is constrained to the UAV autonomy problems of take-off, navigation, and landing.
- Autonomous landing is constrained by fixed position landing platform with ideal operating conditions.
- Autonomous take-off is constrained by taking flight from a fixed position platform, with ideal operating conditions.
- Autonomous waypoint navigation is constrained by absence of obstacles, and operating with ideal operating conditions.

Project Context

- Project will utilize stable ROS distribution
- Project simulations will utilize Gazebo 6.+ & ROS package Rviz
- Project will be developed in Linux environment compliant with ROS & Gazebo

Deliverables

Phase 1

- Requirements documentation
- Overview documentation

Phase 2

- Project software
- Log
- Refence manual (software documentation)
- User documentation
- System design documentation
- Testing documentation
- Deployment documentation

Product Backlog

Phase 1

- **O-1**: As an owner, I want the UAV to autonomously take-off from the landing pad
- **O-2**: As an owner, I want the UAV to autonomously navigate through a series of waypoints

Phase 2

- **U-1**: As a user, I want to communicate the waypoints to the UAV
- **O-3**: As an owner, I want the UAV to autonomously return to the location of the landing pad
- **O-4**: As an owner, I want the UAV to autonomously land on the landing pad without damaging the craft
- **O-5**: As an owner, I want the UAV to autonomously land on the landing pad with the correct orientation

Sprint Report

Completed Tasks

- Install Ubuntu 14.04 or some other ROS Indigo/Jade distro compliant OS.
- Setup Gazebo 6.+
- Download Rviz package
- Review previous iteration of project documentation
- Inspect current quadrotor configuration
- Identify parts needed for quadrotor

Tasks Carried to Next Sprint

- Acquire parts needed for quadrotor

2 Sprint Report #2

Summary

Team Expeditus was able to adapt to setbacks, and make progress on other tasks and goals of the project. Specifically, the team was able to make progress on the landing software, as well as find and implement a visual simulation that models the Pixhawk flight controller. The team was also able to coordinate with our advisor and school faculty for funding and ordering of our UAV platform and components. The restructuring of tasks for Sprint 2, does have a knock-on effect for Sprint 3. Sprint 3 will focus heavily on the building and testing of the UAV and its off-the-shelf components. Additionally, our client/advisor has suggested a different AI approach than the Artificial Neural Network(ANN). We will pursue this development during Sprint 3. Lastly, after meeting with the CENG/EE team several times, it was determined that while there is an opportunity for collaboration, time frames for both teams will not support collaboration. Our team will continue, however, to work with the ME UGV team on the development of a landing pad functional for both teams.

Team Work

- **Julian Brackins:** Worked on tasks relating to Autonomous Landing.
- **Jonathan Dixon:** Worked on tasks relating to Autonomous Landing.
- **Dylan Geyer:** Worked on tasks relating to ordering parts for the UAV,
- **Christopher Smith:** Worked on tasks relating to ordering parts for the UAV, as well as tasks relating to setting up a Simulation Environment.
- **Steven Huerta:** Worked on tasks relating to ordering parts for the UAV, as well as tasks relating to setting up a Simulation Environment.

Completed Backlog

Common Development Tasks

- **Setup Simulation Environment.**
  The team now has a working software simulation of the Pixhawk 4, the flight controller for this build, that utilizes both ROS and Gazebo.
- **Identify Parts Needed for UAV.**
  The team was supplied with funding source. The team needed to additionally coordinate with the SDSMT UAV Team to order correct parts, as well as parts that would be useful to both groups to ensure redundancy in the event of component failure.
- **Acquire parts needed for quadrotor**
  Received approval for the ordering of the parts. Parts ordered. Expected delivery date of 11/9/15.

As a user, I want to communicate the waypoints to the UAV

- **Review code that communicates with quadrotor.**
  Software is available to access the flight controller through a GUI called APM Planner, available for Linux/Windows. Additionally, there is Mission Planner, available for Windows. Both will provide the ability of a user to communicate with the UAV.
- **Review code that allows a user to input waypoints.**
  Both APM Planner and Mission Planner allow the user to input waypoints through the GUI.

As an owner, I want the UAV to autonomously take-off from the landing pad.

- **Review code that enables the quadrotor to autonomously take-off from landing pad.**
  This will be handled by Mission Planner/APM Planner.

As an owner, I want the UAV to autonomously navigate through a set of way-points.

- **Review previous implementation for navigating waypoints.**
  This will be handled by Mission Planner or APM Planner

As an owner, I want the UAV to autonomously return to the location of the landing pad.

- **Review code that allows the autonomous return of the UAV to the landing pad.**
  This will be handled by Mission Planner or APM Planner. The built in autonomy will bring the UAV to a position near the landing pad, where either Visual Homography, Artificial Intelligence, or combination of the two will be responsible for landing the craft. It is estimated that the craft will be within 10 meters of the designated area. Discussions with UAV team members provide an estimate of 5 meters from their observations.

As an owner, I want the UAV to autonomously land on the landing pad without damaging the craft

- **Review previous implementation for autonomous landing.**
  The code was reviewed and is running. The software is correctly identifying the RGB lights and accurately reporting distance.

As an owner, I want the UAV to autonomously land on the landing pad with the correct orientation.

- **Review previous implementation for autonomous landing.**
  As reported above, the software is running and is able to detect the lights. This detection will allow the UAV to orient itself to align correctly with the landing pad.

Uncompleted Tasks

Common Development Tasks

- **Build UAV**
  This will be completed during Sprint 3. Waiting for UAV parts to arrive.
- **Test flight under manual control**
  Testing will be completed by Sprint 3. Waiting for UAV to be built.

Prototype

There is a prototype document for Sprint 2 (found here in the repository), where this same material will be covered in much greater detail. This is only a brief description.

- **Visual Homography Code**
  The Visual Homography Code that was developed last year for the UAV Landing Project has been reviewed. The program successfully builds and much of it is likely to be reused towards providing the landing algorithm. The code can be found here in the repository. The code requires that OpenCV has been installed. A cmake file is contained within the directory, so that after running cmake and make the program can be run by ./tracker. The tracker is looking for RGB blobs, so it may be better for testing to have some primary colors about to test. This program is currently providing correct distance in centimeters.

- **Simulation**
  To test our landing algorithms in simulation, it would be very useful to have something that approximates the Pixhawk flight controller to communicate with for the purpose of supplying instructions to the controller, as well as receiving flight data. The PX4 development team have provided both Software-In-The-Loop and Hardware-In-The-Loop simulation environments. This will require Linux 14.04, ROS Distro Indigo or Jade, and the installation of a few repositories. These are detailed well in the setup document provided by the group here. However, **catkin_ make** command did not build the meta-package correctly, as detailed in the instructions. Rather, **catkin build** will build the meta-package properly and the simulation will run with the assistance of an XBox controller (PS controllers will not work).

- **Ordering Parts**
  Over the course of Sprint 2, the team met with our advisor and faculty for purpose of receiving funding to create a new UAV platform for this project. The team also met with members of the UAV team to receive assistance and guidance in purchasing hardware that would be compatible with UAV team hardware. In the event of a component not functioning, our team would be able to utilize a component from the UAV team. After building a parts list for a hexrotor, we received approval from faculty for our purchase. A complete order list will be detailed in the Prototype document.

3 Sprint Report #3

4 Sprint Report ...

D

Industrial Experience and Resumes

1 Resumes

Your resumes are included here. See the source file (industrial.tex) and uncomment the PDF includes to see how this works. If your resume is written in LaTeX then you can just insert the LaTeX source code.

2 ABET: Industrial Experience Reports

2.1 Name1

2.2 Name2

2.3 Name3

E

Acknowledgment

Thanks

F

Supporting Materials

This document will contain several appendices used as a way to separate out major component details, logic details, or tables of information. Use of this structure will help keep the document clean, readable, and organized.

LaTeX Example

LaTeX sample file: Remove from submitted materials

1 Introduction

This is a sample input file. Comparing it with the output it generates can show you how to produce a simple document of your own.

2 Ordinary Text

The ends of words and sentences are marked by spaces. It doesn't matter how many spaces you type; one is as good as 100. The end of a line counts as a space.

One or more blank lines denote the end of a paragraph.

Since any number of consecutive spaces are treated like a single one, the formatting of the input file makes no difference to TeX, but it makes a difference to you. When you use LaTeX, making your input file as easy to read as possible will be a great help as you write your document and when you change it. This sample file shows how you can add comments to your own input file.

Because printing is different from typewriting, there are a number of things that you have to do differently when preparing an input file than if you were just typing the document directly. Quotation marks like "this" have to be handled specially, as do quotes within quotes: "'this' is what I just wrote, not 'that'".

Dashes come in three sizes: an intra-word dash, a medium dash for number ranges like 1–2, and a punctuation dash—like this.

A sentence-ending space should be larger than the space between words within a sentence. You sometimes have to type special commands in conjunction with punctuation characters to get this right, as in the following sentence. Gnats, gnus, etc. all begin with G. You should check the spaces after periods when reading your output to make sure you haven't forgotten any special cases. Generating an ellipsis . . . with the right spacing around the periods requires a special command.

TeX interprets some common characters as commands, so you must type special commands to generate them. These characters include the following: $ & % # { and }.

In printing, text is emphasized by using an *italic* type style.

A long segment of text can also be emphasized in this way. Text within such a segment given additional emphasis with Roman *type. Italic type loses its ability to emphasize and become simply distracting when used excessively.*

It is sometimes necessary to prevent TeX from breaking a line where it might otherwise do so. This may be at a space, as between the "Mr." and "Jones" in "Mr. Jones", or within a word—especially when the word is a symbol like *itemnum* that makes little sense when hyphenated across lines.

Footnotes[1] pose no problem.

TeX is good at typesetting mathematical formulas like $x - 3y = 7$ or $a_1 > x^{2n}/y^{2n} > x'$. Remember that a letter like x is a formula when it denotes a mathematical symbol, and should be treated as one.

[1] This is an example of a footnote.

3 Displayed Text

Text is displayed by indenting it from the left margin. Quotations are commonly displayed. There are short quotations

> This is a short a quotation. It consists of a single paragraph of text. There is no paragraph indentation.

and longer ones.

> This is a longer quotation. It consists of two paragraphs of text. The beginning of each paragraph is indicated by an extra indentation.
>
> This is the second paragraph of the quotation. It is just as dull as the first paragraph.

Another frequently-displayed structure is a list. The following is an example of an *itemized* list.

- This is the first item of an itemized list. Each item in the list is marked with a "tick". The document style determines what kind of tick mark is used.
- This is the second item of the list. It contains another list nested inside it. The inner list is an *enumerated* list.
  1. This is the first item of an enumerated list that is nested within the itemized list.
  2. This is the second item of the inner list. LaTeX allows you to nest lists deeper than you really should.

  This is the rest of the second item of the outer list. It is no more interesting than any other part of the item.
- This is the third item of the list.

You can even display poetry.

> There is an environment for verse
> Whose features some poets will curse.
>
> For instead of making
> Them do *all* line breaking,
> It allows them to put too many words on a line when they'd rather be forced to be terse.

Mathematical formulas may also be displayed. A displayed formula is one-line long; multi-line formulas require special formatting instructions.

$$x' + y^2 = z_i^2$$

Don't start a paragraph with a displayed equation, nor make one a paragraph by itself.

4 Build process

To build LaTeX documents you need the latex program. It is free and available on all operating systems. Download and install. Many of us use the TexLive distribution and are very happy with it. You can use a editor and command line or use an IDE. To build this document via command line:

```
alta>  pdflatex SystemTemplate
```

If you change the bib entries, then you need to update the bib files:

```
alta>  pdflatex SystemTemplate
alta>  bibtex SystemTemplate
alta>  pdflatex SystemTemplate
alta>  pdflatex SystemTemplate
```

The template files provided also contain a Makefile, which will make things much easier.

Acknowledgment

Thanks to Leslie Lamport.