**Graduate Projects**

University of Colorado at Boulder

Aerospace Engineering Sciences

ASEN 5018/6028 –Fall 2015

|  |
| --- |
| **FlyNet**  **Planning Subsystem Summary/Continuity Document** |

|  |  |  |
| --- | --- | --- |
| **Approvers List** | | |
|  | Title | Name |
| Prepared By | Planning Team | Bryce Hill |
|  |  | Edward Meletyan |
| Approved By | Planning Lead | Bryce Hill |
| Approved By | Systems Engineer |  |
| Approved By | Project Manager |  |

**1: Introduction & Summary**

To complete our objective of searching a space for targets a path must be generated which: avoids obstacles, arrives at desired final destination and is possible to compute in real time environments.

The operational space is 3-dimensional. However, 3D planning algorithms were observed to require more than twenty seconds to compute. Therefore, the decision was made to use 2-dimensional algorithms which can be computed in milliseconds.

Two algorithms have been evaluated: the first is a hybrid Voronoi/A\* method which uses the Voronoi algorithm to generate points between obstacles and the A\* algorithm to plan the most efficient path from start to goal. The second path planning algorithm is a probabilistic road map (PRM) approach. The PRM creates a random sample of nodes on a map that are located in free space (i.e. not an obstacle). Each node is connected to every neighboring node that is within a configured threshold until a path can be made from a starting point to a goal point based on the shortest distance between connected nodes.

# **2: Semester Report**

## 2.1: Objectives and Tasks List

**Completed**:

1. Matlab implemented Voronoi/A\* 2D path planning algorithm
2. Matlab implemented probabilistic road map 2D path planning algorithm
3. Designed a perpetual motion machine

**Incomplete**:

1. Test each planning algorithm in Matlab simulation

## 2.2: Issues

1. Testing the planning algorithms in the Matlab simulation remains incomplete due to time constraints.

## 2.3: Lessons Learned

So you just identified some things that prevented you from completing some tasks – is there anything you can learn from that? Having more time to do things isn’t really a lesson learned, so anything like Issue 3 in my example above won’t really have a lesson learned. Remember: lessons learned are anything that you **wish you knew before**, something **learned the hard way**, or a **piece of advice**. Note that you don’t have to address each numbered issue above directly. Also note that you are encouraged to include lessons learned in the process of completing the tasks that you *did* finish.

1. Be very clear on the requirements and their implication prior to beginning implementation.

## 2.4: Procedures

Voronoi/A\*:

We began with Matlab’s built in Voronoi function. Voronoi provides a convenient way to ensure paths are equidistant between obstacles. In Figure 1 below, obstacles are represented by the ‘\*’ and the paths are drawn between vertices which the voronoi algorithm returns. These points are processed and a matrix holding the indices of neighboring nodes is constructed. Additionally, if paths are too close to obstacles they will be marked out of bounds.



Figure . Voronoi generated paths

Next, A\* was implemented in Matlab. A\* plans the path of nodes which will be navigated to reach the goal. In the following, Figure 2, the origin is represented as a circle, the goal as X. The paths with dashed lines (that aren’t red) have been eliminated due to their proximity to obstacles.



Figure . Voronoi/A\*

Probabilistic Roadmap (PRM)

The PRM approach requires an initial obstacle map, starting coordinates, and goal coordinates. The obstacle map is a two-dimensional array containing ‘0’ for coordinates that are empty space and ‘1’ for coordinates that are obstacles. Given this information, the PRM algorithm creates a random set of nodes in empty space.



Figure . Probabalistic Road Map

Table 2.: Software list

|  |  |  |
| --- | --- | --- |
| Program Name | Version | Purpose |
| Matlab | R2015a | Algorithmic development and testing |
|  |  |  |

# **3: Next Semester/Future Expectations**

## 3.1: Prioritized List of Tasks and Objectives

Include your incomplete tasks ***and*** next steps for your subsystem. It is important for you to be thinking ahead. If your project is not continuing next semester, summarize what you think could be done if your project was reinstated in the future.

Accomplished

1. Evaluate 3D vs. 2D algorithms
2. Develop PRM and Voronoi/A\* algorithms in Matlab

Next Steps

1. Integrate algorithms with Matlab vehicle simulation
2. Convert algorithms to C
3. Integrate with onboard computer

## 3.2: Starting Points

For each one of the tasks and objectives in 3.1, describe to the best of your knowledge where the person reading this can start. The person reading this is either going to be you after over a month off, or someone totally unfamiliar with how to go about things. **Point to the locations in the server where you have pertinent files saved.** Now the person has the file open in whatever software, what should they work on doing to start with?

1. Structural design of the toaster can be found on the project server under Working Directories > By Subsystem > Structures > CAD > Toaster.cad. We suggest first adding a best estimate for the heating coils to place volume constraints on the real coils that can be used. Other work that needs to be done is ensuring the polish will be shiny enough to see your face in the reflection.
2. The lingering issue in finalizing the peanut butter – jelly interface is the type of jelly to be used. We have collected some data on various jellies, but a trade study should be performed ASAP to determine the best jelly option.
3. Fish sticks have already been chosen but their crumbliness has not yet been analyzed. We suggest importing the CAD model of the fish stick and performing FEA stress analysis on the fish sticks to determine how much mass will be lost to rogue crumbs.

## 3.3: Improvement, Updates, Verification

For tasks you have completed, what could/should be done to improve or update them in the future? Here is a good place to blatantly state all the assumptions you have made, and prioritize them in order of the impact the assumption has on your result. As assumptions later get filled with more concrete data, your analysis will need to be updated and/or verified to ensure no issues have been raised.

Note: Be careful with improvements -- remember the goal is always to meet the requirement and not go any further.

1. We have assumed that there will be a large body of water near Boulder, CO for the team yacht. When that assumption is verified, the yacht should be moved to the body of water. Should this assumption prove to be invalid, drastic redesign of the boat may be necessary.
2. We have assumed that the question “what is the meaning of life” is one that makes sense. We have an answer to this question, but the question may need to be updated in the future as it may be out of the scope of this project.
3. In the design of the perpetual motion machine, we assumed the existence of tachyon particles to provide superluminal ghost-forcing on our gyroscopic, electromagnetic tether. Specifically, the tachyons are expected to produce a force of 30N. This number was determined using superluminal, reverse-time Feynman diagrams but should be updated when a better understanding of the interaction between the Higgs boson, muons, and tau-neutrinos is obtained.