8-2008

Using Automated Task Solution Synthesis to Generate Critical Junctures for Management of Planned and Reactive Cooperation between a Human-Controlled Blimp and an Autonomous Ground Robot

Christopher M. Reardon
University of Tennessee, Knoxville

To the Graduate Council:

I am submitting herewith a thesis written by Christopher M. Reardon entitled "Using Automated Task Solution Synthesis to Generate Critical Junctures for Management of Planned and Reactive Cooperation between a Human-Controlled Blimp and an Autonomous Ground Robot." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Lynne E. Parker, Major Professor

We have read this thesis and recommend its acceptance:

Bradley Vander Zanden, Dongjun Lee

Accepted for the Council:
Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a thesis written by Christopher M. Reardon entitled “Using Automated Task Solution Synthesis to Generate Critical Junctures for Management of Planned and Reactive Cooperation between a Human-Controlled Blimp and an Autonomous Ground Robot.” I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Dr. Lynne E. Parker, Major Professor

We have read this thesis
and recommend its acceptance:

Dr. Bradley Vander Zanden

_____

Dr. Dongjun Lee

_____

Accepted for the Council:

Carolyn R. Hodges,
Vice Provost and
Dean of the Graduate School

(Original signatures are on file with official student records.)

Using Automated Task Solution Synthesis to Generate Critical Junctures for Management of Planned and Reactive Cooperation between a Human-Controlled Blimp and an Autonomous Ground Robot

A Thesis
Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Christopher M. Reardon
August 2008

Abstract

This thesis documents the use of an approach for automated task solution synthesis that algorithmically and automatically identifies periods during which a team of less-than-fully capable robots benefit from tightly-coupled, coordinated, cooperative behavior.

I test two hypotheses: 1) That a team's performance can be increased by cooperating during certain specific periods of a mission and 2) That these periods can be identified automatically and algorithmically. I also demonstrate how identification of cooperative periods can be performed both off-line prior to the application and reactively during mission execution.

I validate these premises in a real-world experiment using a human-piloted Unmanned Aerial Vehicle (UAV) and an autonomous mobile robot. For this experiment I construct a UAV and use an off-the-shelf robot. To identify the cooperative periods I use the ASyMTRe task solution synthesis system, and I use the Player robot server for control tasks such as navigation and path planning.

My results show that teams employing cooperative behaviors during algorithmically identified cooperative periods exhibit better performance than non-cooperative teams in a target localization task. I also present results showing an increased time cost for cooperative behaviors and compare the increased time cost of two cooperative approaches that generate cooperative periods prior to and during mission execution.

Table of Contents

Table of Figures

1. Introduction

In modern AI robotics, there is a strong interest in developing teams of robots capable of performing specific applications [18]. This is in part because the complexities of real-world applications often exceed the abilities of individual robots.

There is also a well-reasoned desire to implement teams of heterogeneous robots [12][13][14][17][19]. Frequently, a team of robots with varying abilities can match or exceed the performance of a single, general-purpose robot, and can have many other benefits as well. Teams can be heterogeneous and can have robots with specialized equipment for performing specific tasks, and robots can be of varying sizes, shapes, and abilities. They can also have multiple robots with the same capabilities for redundancy. Cost can also be a consideration, as multiple less-capable robots can cost less than a single, more-capable robot.

From the perspective of an application, many tasks can benefit from or even require multiple simultaneous actions performed by different team members. Coordination or cooperation between team members can yield an increase in performance efficiency for these applications [13][19].

In many situations, there is the opportunity or desire for humans to interact with or control members of a team of robots. Human-controlled and autonomous robots have inherently different capabilities. For example, human-controlled robots can be much more adaptively controlled, and autonomous robots are often more precise and perform better at repetitive tasks. Human interaction can also allow for a degree of fault tolerance and detection greater than that of a completely autonomous system.

While cooperation can have great benefits, it is not difficult to conceive of an application where each robot's capabilities are not required to perform all of the aspects of a task at all times. In such an application, a team of robots with different capabilities may cooperate to work together to increase overall performance, but while tightly-coupled cooperation will benefit overall performance, such cooperation is not necessary at all times. It is possible to identify those periods where tightly-coupled cooperative behavior would benefit the team's performance and those periods where a non-cooperative or independent behavior would best benefit overall team performance. Previous work has undertaken the task of manually identifying such periods of beneficial cooperation [8].

Previous work has also been undertaken by Parker and Tang [20][21] to algorithmically and dynamically generate team configurations based on information types and motor and perception schemas available at the time using a system called ASyMTRe. ASyMTRe is a reasoning system that maps available information types to sensors and perceptual and motor control schemas to synthesize robot team configurations and behaviors for a task [20][21]. In this project I make use of the ASyMTRe system to algorithmically identify those periods during which tightly-coupled coordination/cooperation benefits a team of less-than-fully-capable robots.

I then show, using a real-world team of autonomous and human-controlled robots, an increase in efficiency over independently-operating teams of robots when "synergistically" cooperative behaviors are employed. Additionally, I show that by using ASyMTRe, I can both generate the periods for employing these cooperative behaviors

prior to the beginning of the application and reactively, during the execution of the application.

1.1 Contributions

In previous related work [8], cooperative periods were identified manually in an offline process prior to an application, with validation of the approach done only in simulation. The contributions of my thesis are the identification of cooperative periods in an algorithmic method, rather than manually, both a priori offline and reactively online, and the validation of this concept of cooperative periods online, through real-world experimentation, rather than in simulation, using a human-controlled UAV and an autonomous ground robot.

2. Related Work

Extensive work in human-robot interaction has been conducted, as referenced by Tang and Parker in [22]. As the authors discuss, there is a strong motivation for "peer" interaction between robots and humans rather than human supervision of robots. Motivations include the ability to perform more complex tasks, better fault tolerance, and better team autonomy [16]. Furthermore, when humans are included as team members, how to model human capabilities becomes an area of special consideration. Some work takes a full-featured approach, for example, in [10], a cognitive architecture model of human capabilities is created. For this thesis, no such complex modeling is necessary. I take a more minimalistic approach and only define the information necessary and required abilities available for the generation of a task solution.

My thesis makes use of previous related work called Automated Synthesis of Multi-robot Task solutions through software Reconfiguration, or ASyMTRe. The following paragraphs paraphrase (with permission of the authors) related work [22] by Parker and Tang, the creators of ASyMTRe:

The ASyMTRe approach was developed for addressing the formation of heterogeneous robot coalitions that solve a single multi-robot task. More generally, the approach deals with the issue of how to organize robots into subgroups to accomplish tasks collectively based upon their individual capabilities.

The fundamental idea of ASyMTRe is to change the abstraction that is used to represent robot competences from the typical "task" abstraction to a biologically-inspired "schema" [7][15] abstraction and providing a mechanism for the automatic reconfiguration of these schemas to address the multi-robot task at hand. In the ASyMTRe view, robot capabilities are a set of environmental sensors that are available for the robot to use, combined with a set of perceptual schemas, motor schemas, and communication schemas that are pre-programmed into the robot at design time.

The ASyMTRe approach extends the prior work on schema theory by autonomously connecting schemas at run time instead of using pre-defined connections. According to information invariants theory [9], the information needed to activate a certain schema or to accomplish a task remains the same regardless of the way that the robot may obtain or generate it. We can label inputs and outputs of all schemas with a set of information types, for example, laser range data, global position, etc. Two schemas can be connected if their input and output information labels match. Thus, schemas can be connected within or across robots based upon the flow of information required to accomplish a task. With the run time connection capabilities, task solutions can be configured in many ways to solve the same task or reconfigured to solve a new task. Additionally, more capable robots can share information to assist less-capable robots in accomplishing a task.

Parker and Tang concluded that the ASyMTRe approach provides mechanisms for multiple robots to (1) synthesize task solutions using different combinations of robot sensors and effectors, (2) share information across distributed robots and form coalitions as needed to assist each other in accomplishing the task, and (3) reconfigure new task solutions to accommodate changes in team composition and task specification.

This thesis also builds on the previous concepts detailed in an unpublished internal paper from Lockheed Martin Advanced Technology Laboratories, "Distributed Control for Unmanned Vehicles," by Choxi and Bolden [8]. In it, Choxi and Bolden investigate tightly-coupled coordination between a team of a UAV and a ground robot and identify the concept of "critical junctures" (CJs): the "points in a mission where (independent) behaviors will fail and tightly coupled coordination is required" and "points where tightly-coupled coordination is no longer required."

My effort here is to expand this work in two main directions. First is to validate the concept of using critical junctures to identify periods best suited for cooperation/coordination and independent behaviors. Because the previous authors did their work in a simulated environment, I wish to perform this validation in a real-world experiment environment. Second is to use the ASyMTRe task solution synthesis system for the three purposes mentioned previously from [22]: to synthesize task solutions that, where necessary, create a coalition robot team to assist each other and share information and to reconfigure the task solution, in my case in response to changes in available environmental information and perceptual schemas. In that way, I use ASyMTRe to autonomously generate critical junctures by identifying the points at which the task solution must be reconfigured.

This allows the identification of critical junctures to be autonomous, for example if done off-line prior to the mission. It also allows for the performance of the task that Choxi and Bolden refer to as "distributed planning", which enables what I refer to as reactive cooperation – the ability to identify critical junctures on-line during the course of a mission and adapt the plan accordingly.

3. Approach

The approach to this project is described in two main parts. First, the conceptual approach to using ASyMTRe to generate critical junctures is explored; then, the experimental approach details how I set about testing the concepts.

3.1 Conceptual Approach

The conceptual approach is split into the topics of critical junctures and ASyMTRe and the concepts behind the behaviors I employ in my approach.

3.1.1 Synthesizing Critical Junctures

The complexities of real-world applications often exceed the abilities of an individual robot. Frequently, a team of robots with varying abilities can match or exceed the performance of a single, all-purpose robot, and can have other advantages, such as cost, redundancy, multiple points of failure, speed, etc. By creating a heterogeneous team of robots to perform a task, performance efficiency can sometimes be increased. Also, in many applications not all of a robot's capabilities are required to perform all of the aspects of a task at all times. In such an application, a team of robots with different capabilities may cooperate to work together at some points where team performance would be benefitted and work separately where no cooperative benefit is realized.

These points, at which robot team members can work together to benefit team performance on a particular task have been called "critical junctures", or "CJs" [8]. Likewise, points at which cooperating team members can stop working together and not adversely impact performance have also been dubbed critical junctures; thus, critical junctures define the boundaries of the period during which multiple robots on a team can cooperate to increase efficiency in the performance of a task.

Critical Junctures and ASyMTRe

While previous work has been performed that validates the concept of CJs for a task and that cooperation during the critical period bounded by CJs increases team performance, such a cooperative period has been created by identifying the CJs manually and a priori, and validation was conducted in simulation [8].

The ASyMTRe system uses information types, including robot control/behavioral information, perceptual information, and environmental information, as well as perceptual and motor schemas, to dynamically configure robots and teams of robots for the performance of a task.

The benefit of leveraging the power of ASyMTRe to identify critical junctures is twofold: first, it allows us to take the human out of the equation and generate CJs algorithmically, and second, it means the CJs can be generated both as part of an a priori planned path, as in previous work, and reactively, on-the-fly during the execution of an application.

To use ASyMTRe in this manner to identify critical junctures, it is necessary to define the task, information types and schemas in such a way that there can be regions of

beneficial cooperation, but not all aspects of the task require cooperation. It is also highly desirable for this definition to be realistic and generalizable.

Environmentally Dependent Information

In order to utilize ASyMTRe, the information types and schemas available must first be defined. For my purposes, it was sufficient to adapt the information types and schemas defined in previous works [20] and augment those information types with a new "Environmentally Dependent Information" type, or EDI. The concept of an EDI is straightforward: specific schemas available to a robot are dependent on the immediate environment around the robot. In my case, I defined a "landmark" EDI, near which the localization schema was available to one team member, an Unmanned Aerial Vehicle (UAV) robot, and away from which the localization schema was not available to the UAV (Figure 1). Of the previously defined information types and schemas, the UAV retained those that imparted vision-based localization relative to another robot; e.g., given a communicated global position of another robot in view, a robot can determine its own global position by calculating its position relative to the other robot. Thus, I specify regions of the environment where a UAV can localize on its own and others where a UAV can be assisted by a robot in localization.

The new ASyMTRe configuration files for within range of an EDI are shown in Appendix A-1 and A-2. With the configuration of ASyMTRe complete, I am able to use it to identify the CJs for a path both in a pre-planned manner and reactively, during experimentation.

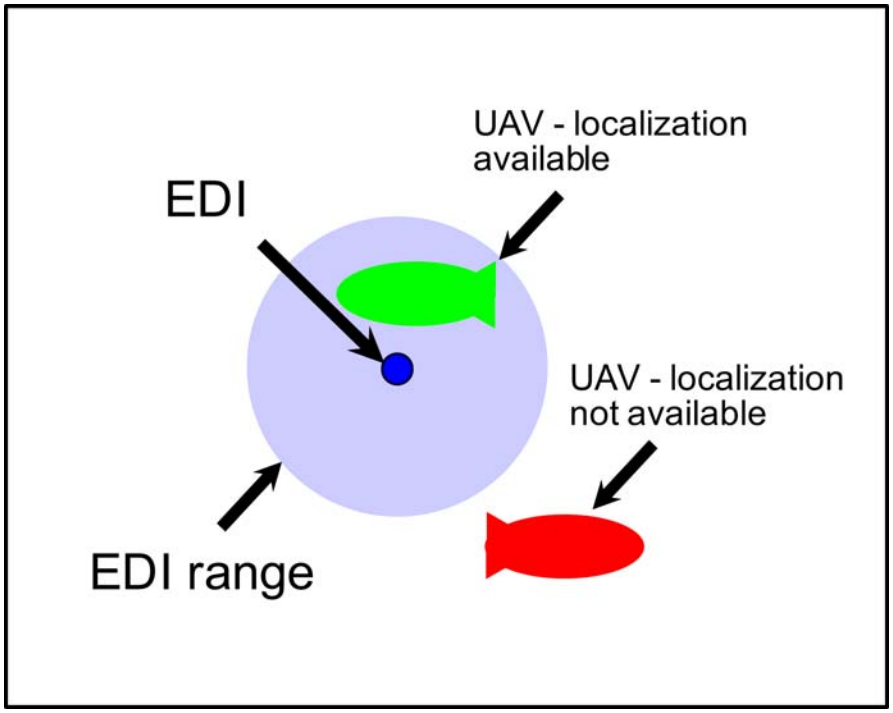

Figure 1: Environmentally Dependent Information Type, (EDI) "Landmark" Concept. When the UAV is within a specified range of an EDI, localization schemas are available.

This CJ identification can be performed by discretizing the robot and UAV paths into two series of points and checking the task solution generated by ASyMTRe at each point. A different set of configurations is used depending on whether the point is within or is not within range of an EDI. When the generated task solution changes between an independent solution and a cooperative solution (i.e., assisted localization), that point is identified as a critical juncture.

3.1.2 Behaviors

To validate the automated selection of CJs, three scenarios are devised for real-world experimentation. In each, a ground robot and a UAV are used, just as in [8]. Each scenario represents a different approach to the problem of identifying Critical Juncture points. The three scenarios are Independent, Cooperative Planned, and Cooperative Reactive, and are outlined in the following paragraphs:

Independent

In the "Independent" behavior scenario, the UAV and robot each perform the task independently, without any cooperation. This represents a "base case" for comparison where no CJs are identified, so no cooperation takes place.

Cooperative Planned

In the Cooperative Planned scenario, the CJs for the robot and the UAV's cooperation are planned in advance of the start of the scenario, as in [8]. Conceptually, the CJs define any period that team members could work together to accomplish any task. A few examples of such cooperation could be localization, box pushing, or assisted navigation. The CJs for this scenario are still algorithmically generated using ASyMTRe; they are simply generated a priori and offline.

Cooperative Reactive

In the Cooperative Reactive scenario, the CJs where the robot and the UAV cooperate are identified (algorithmically, using ASyMTRe) as needed during experiment runtime, in "reaction" to the information types and schemas available.

3.2 Experimental Approach

In order to test the conceptual approach in a real-world environment, an experimental approach is required. The UAV is piloted by a human, and the ground robot is autonomous.

3.2.1 Task

The task I use to test the cooperative behaviors is a coverage problem, i.e., to visit all of the space looking for targets. This task is performed by both the robot and the UAV. Both robot and UAV are provided a map of the environment a priori. Both also have a predefined path to follow through the environment.

The robot, at predefined waypoints along the path, stops and conducts a 360° visual sweep of the area with the camera/blobfinder, looking for targets. The UAV,

being less-precisely controllable than the robot, follows the predefined path and, as deemed necessary by the piloting team, rotates and changes altitude to visually scan the surrounding area for targets.

Upon detecting a target, the robot or UAV then localizes the target to the best of its ability. For the robot, this involves using the direction and size of the target "blob" combined with self-location and orientation (provided by the Player [2][11] localization interface) to determine a precise X,Y position of the target. Because of the robot's height and camera capabilities, all Z-axis values reported by the robot are considered at ground level, or "low." For the UAV, target localization involves a human operator using self-localization, determined by position and orientation relative to Environmentally Dependent Information or the robot, combined with direction and size of target, to visually estimate and record the position of the target. The human operator identifies the X, Y position as well as a Z value of "high" or "low" per target.

Because of the way the targets are distributed and the information types defined for the environment, the UAV and robot are expected to have different levels of success localizing targets for different target locations. For example, targets placed at a high elevation in the environment are unlikely to be detected by the robot, whereas targets in areas outside the range of EDIs are unlocalizable to the UAV without assistance.

In this experiment, while there are targets that are detectable only by the UAV, there are no targets only detectable by the robot. While in this special case a solution could be designed where the robot is dedicated to assisting the UAV, such a solution would work only for this case. The solutions described here are generalizable and as such will work in the case where there are targets localizable only by the robot; therefore, I present and evaluate the solutions in that context.

3.2.2 Targets

For the search task, many criteria influenced the selection of the target item. In order to be easily perceivable by the simple blobfinder proxy, the targets should be of one solid color that is detectable by the blobfinder and reasonably unique in the environment. The targets should present the same two-dimensional profile when viewed from any angle. They must be large enough so that they can be distinguished from small abnormalities in the environment and so that relative distance from the target can be easily determined, while still being small and light enough to safely mount on walls and ceilings. 12" balloons were selected for their nearly-spherical shape, controllable size, light weight, and solid color. They are also inexpensive and require no preparation other than inflation. Their only negative quality is that deflated balloons need to be re-inflated or replaced in between experimentation days. Careful testing revealed that green balloons are best in terms of being distinguishable from other objects in the environment and identifiable in varying light conditions. A target in the environment is shown in Figure 2.

For this project a program was written that takes for input the laser map of the environment, as well as desired target count and spacing. From all eligible positions in the test area, a specified number of positions are randomly generated with at least the minimum distance specified in between each target. No maximum spacing is specified.

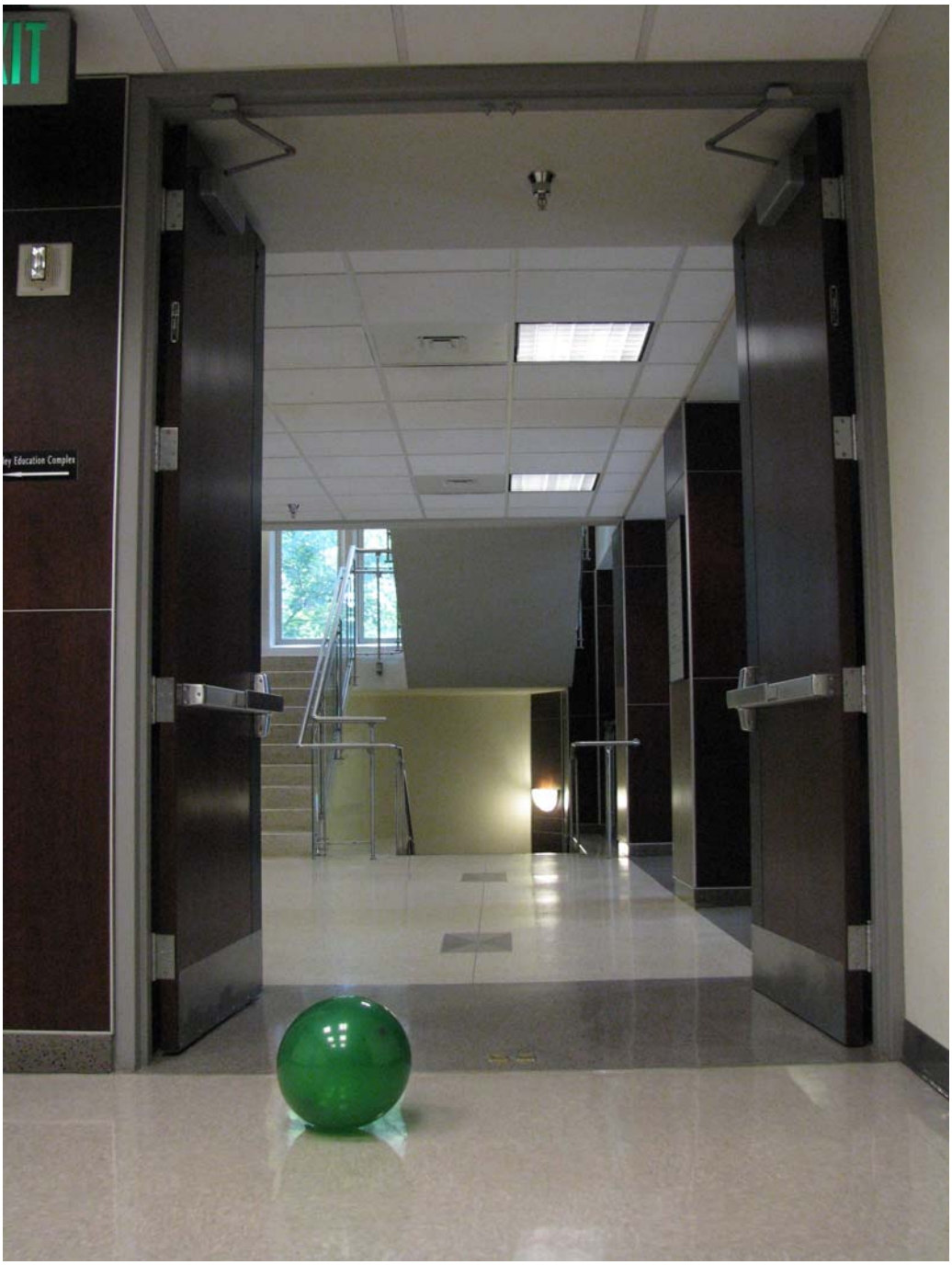

Figure 2: A target (green balloon) in the test environment.

The height of the targets (either high or low) is also selected randomly from eligible positions. If no arrangement that satisfies all constraints is found in a large number of iterations (around 10,000), the script restarts. If no configuration is found in 30 seconds, the script ends with an error indicating that given the number of targets and the required minimum spacing, no configuration is possible. If the script successfully chooses all of the targets, it outputs the position coordinates in number form as well as graphically by editing the input map, as in Figures 3 – 7.

Target spacing values between 1m and 5m were tested before an inter-target spacing of 4m was selected. 4m was deemed reasonable as it has the advantage of being twice the (2m) error of the robot's blobfinder-based target localization, which should help to avoid some correspondence problem issues, while still being close enough together to allow an interesting variance in the target distribution. Given the size of the target-eligible area, a target count of 12 was selected, as experimental testing showed that more than 14 targets could not fit in the eligible area at 4m minimum spacing, and 12 targets allowed for a large variety of target distribution configurations.

Before each experiment, targets were deployed throughout the environment by volunteers. Care was taken to ensure that, if possible, the person performing the UAV target identification/localization task did not participate in the target deployment and thus had no prior knowledge of the targets' locations.

3.2.3 Environment

In preparation for experimentation, a laser map of the experiment area was autonomously constructed by a Pioneer robot using Player and Pmap [6] (Figure 8).

For the experiment area, a portion of the northern end of the second floor of the Claxton Education building was chosen (Figures 9 and 10). The experiment area has the benefits of portions with high ceilings, wide hallways, and tables, benches, chairs, etc. that provide a widely varied indoor environment. The experiment area is approximately 400-500 m^2. For the experiments, tables and benches in the environment are draped with fabric and tablecloths to give them a larger and clearer laser signature. Tables in the vending machine / eating area are arranged to create an area (approx. 40 m^2) inaccessible to the robot but accessible to the UAV. Additionally, tables and benches are arranged in the lobby area to provide a more diverse environment while allowing room for robot navigation throughout.

Correspondence Problem

The search method employed by the robot involves stopping at intervals and scanning 360° for a target. Additionally, all targets are identical in appearance; they are not distinguishable from each other in any way except location. It is not difficult to conceive of a situation where the same target could be detected during the scan at each waypoint. For example, if a target is between two waypoints, it could easily appear in the target scans at both waypoints. Also, I experimentally calculated the error for target localization to be <= 2m. Thus, very rarely does the robot provide an exactly correct localization of a target, and a single target can be localized multiple times. This introduces a problem of identifying the correspondence between the actual target position

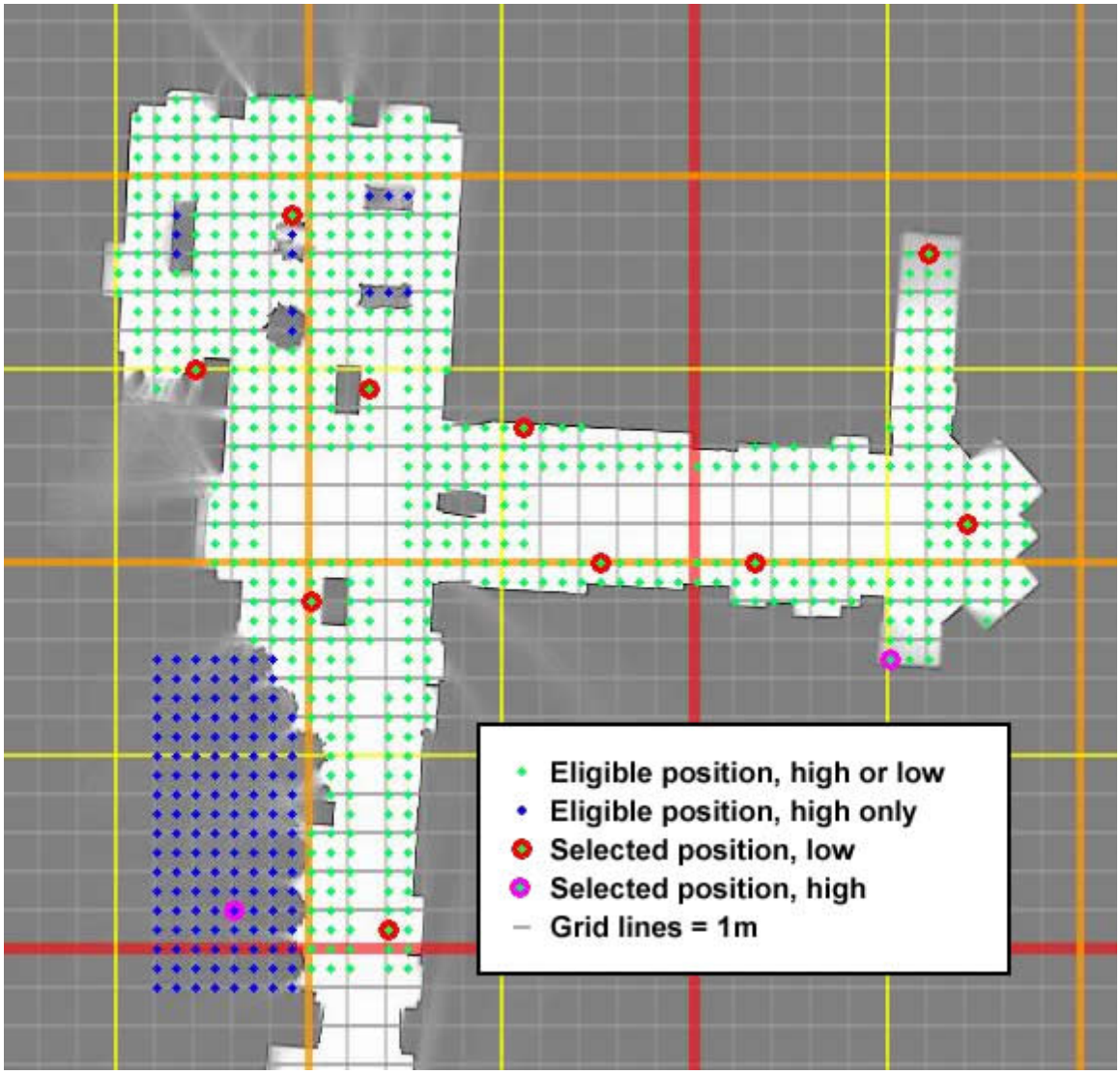

Figure 3: Map 1 – Target locations. Small green dots represent eligible positions, either high or low. Small blue dots represent eligible positions, high only. Red circles are high target locations. Purple circles are low target locations.

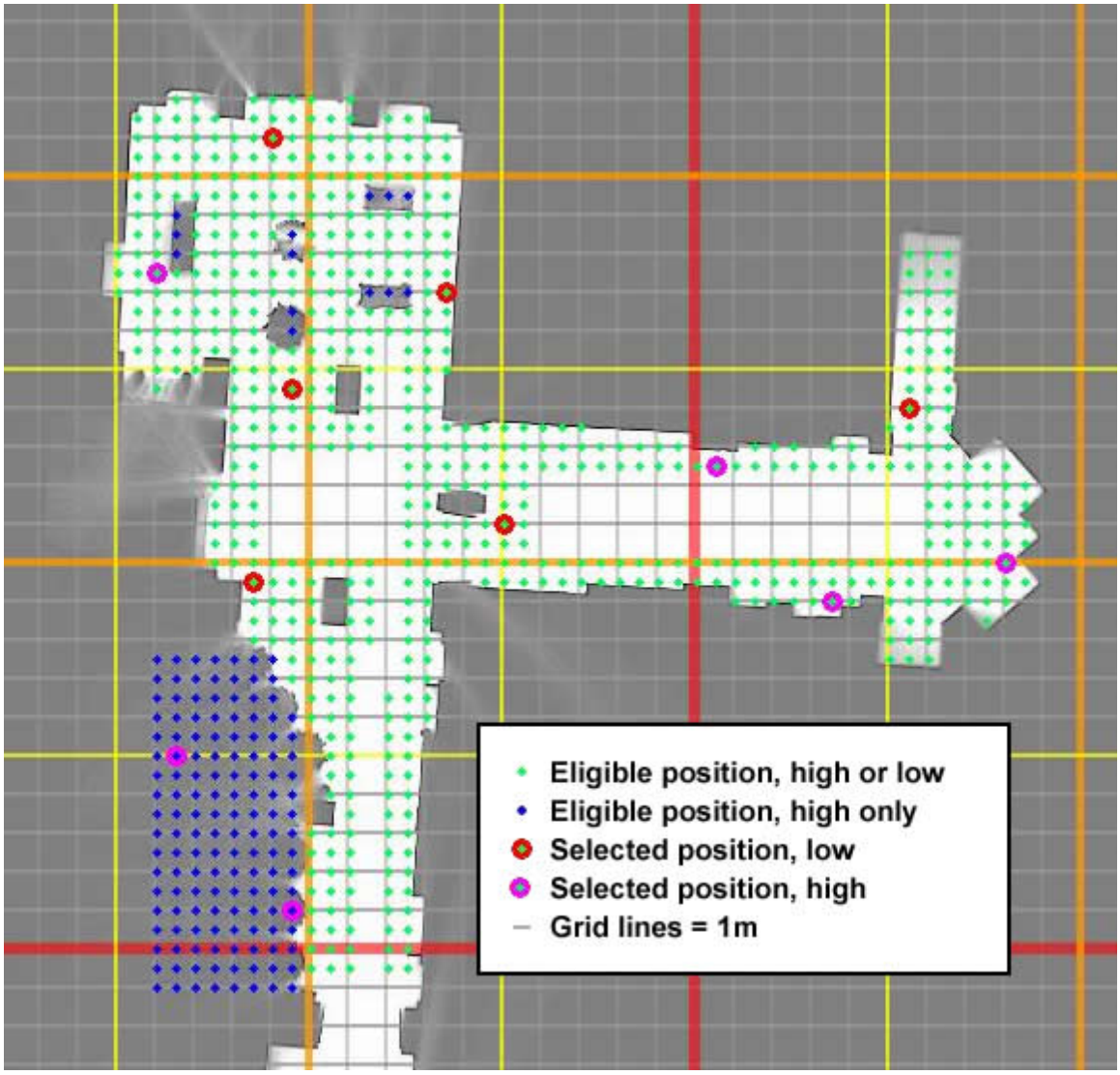

Figure 4: Map 2 – Target locations Small green dots represent eligible positions, either high or low. Small blue dots represent eligible positions, high only. Red circles are high target locations. Purple circles are low target locations.

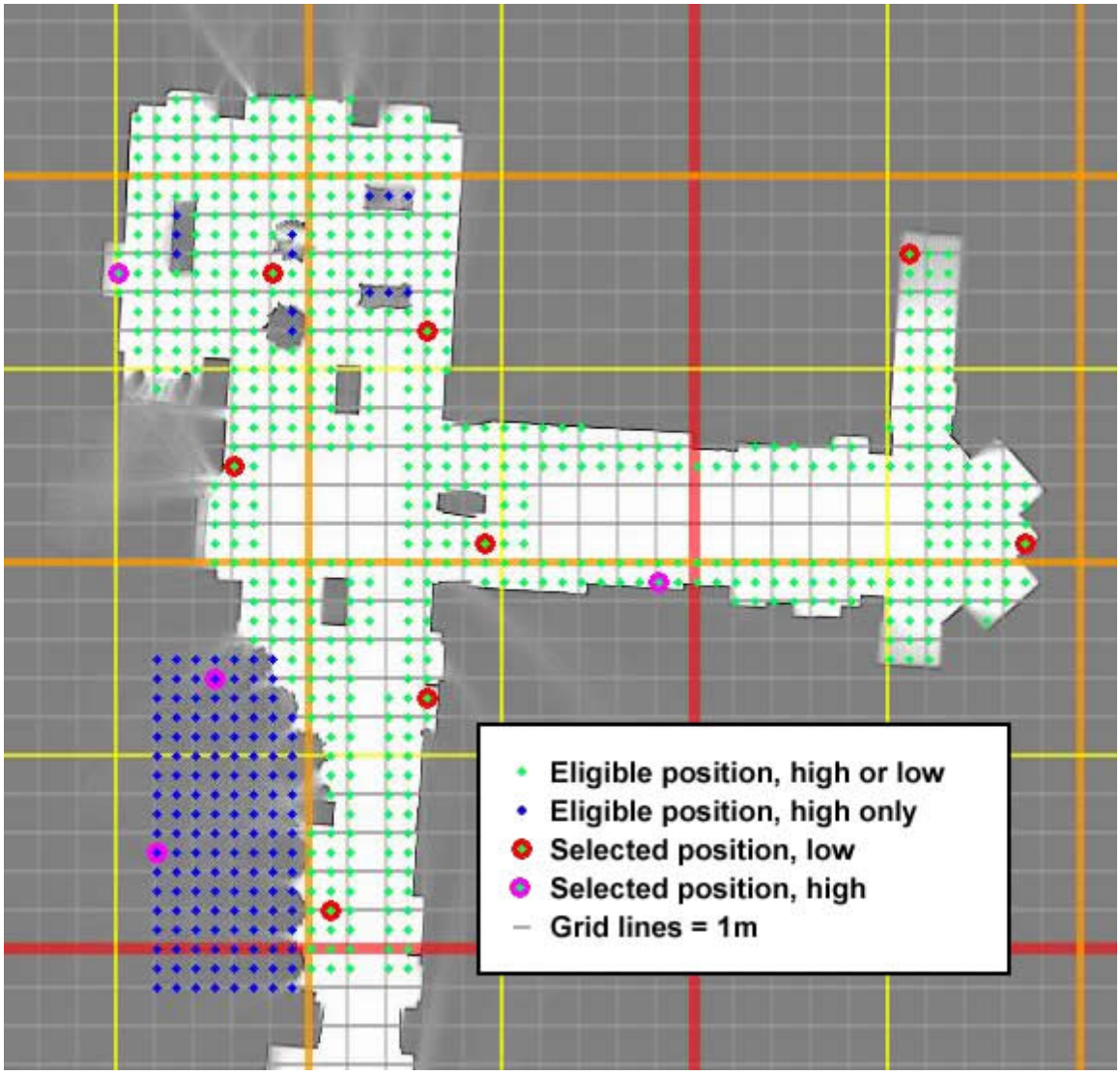

Figure 5: Map 3 – Target locations. Small green dots represent eligible positions, either high or low. Small blue dots represent eligible positions, high only. Red circles are high target locations. Purple circles are low target locations.

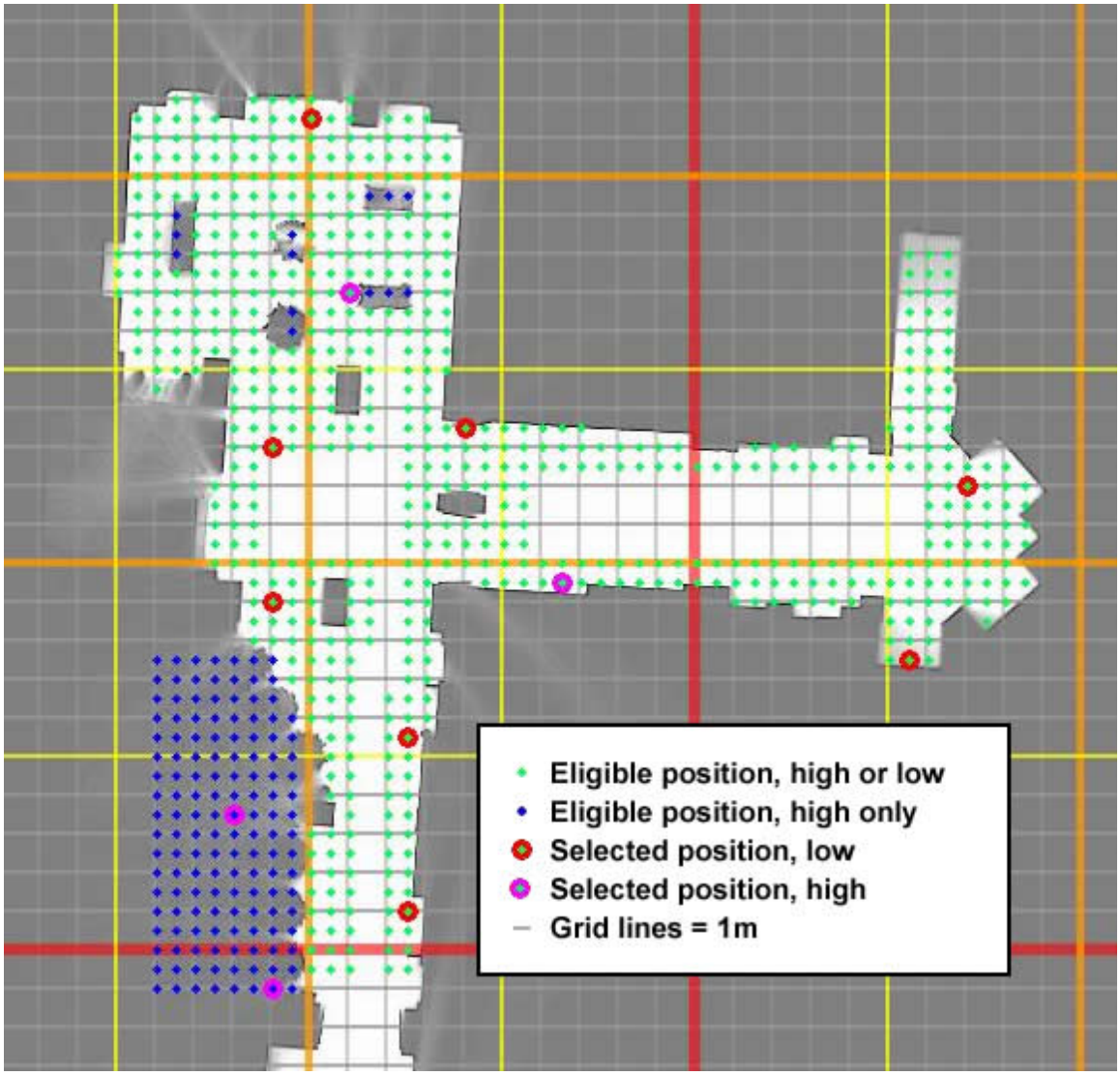

Figure 6: Map 4 – Target locations. Small green dots represent eligible positions, either high or low. Small blue dots represent eligible positions, high only. Red circles are high target locations. Purple circles are low target locations.

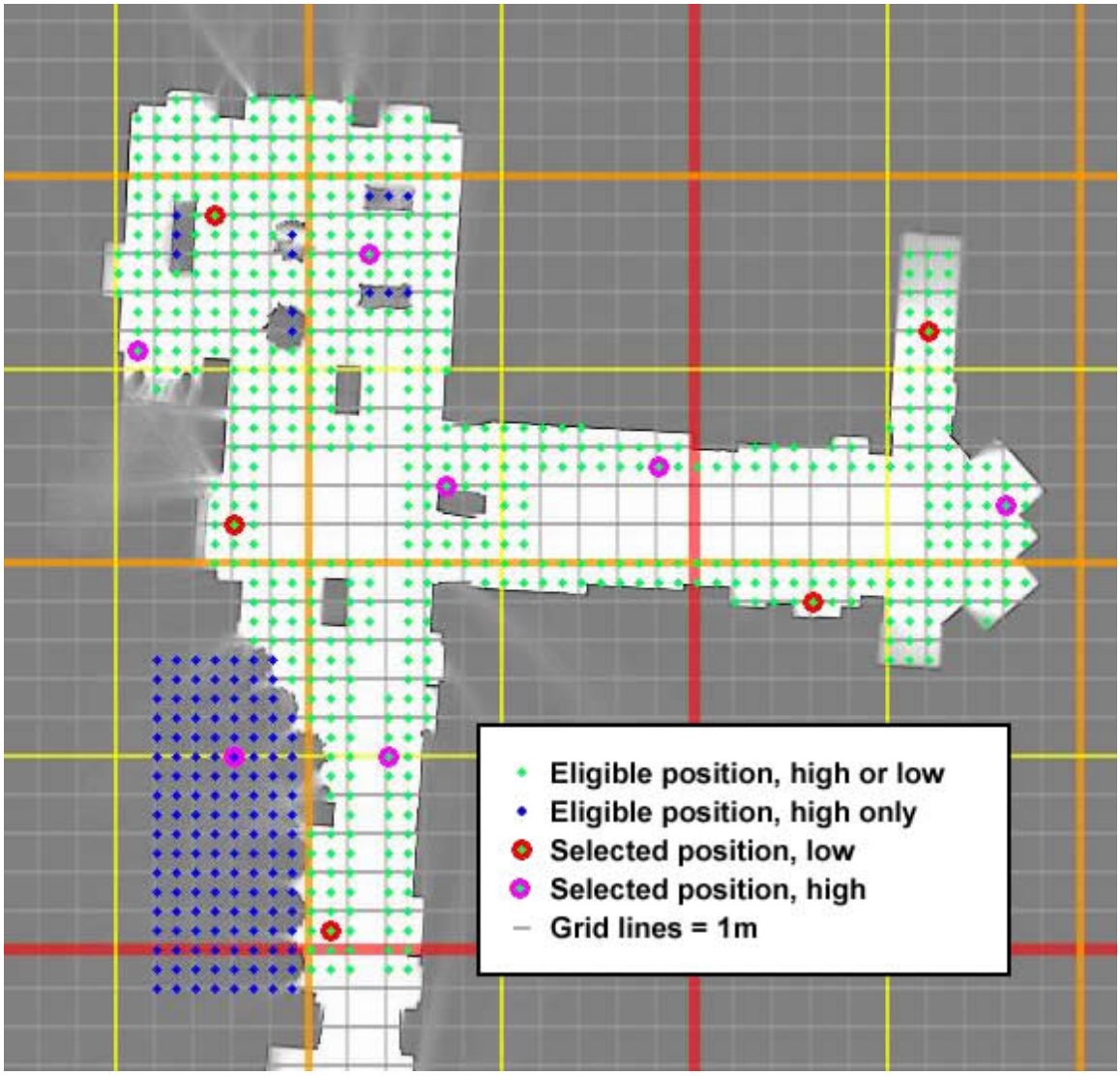

Figure 7: Map 5 – Target locations. Small green dots represent eligible positions, either high or low. Small blue dots represent eligible positions, high only. Red circles are high target locations. Purple circles are low target locations.

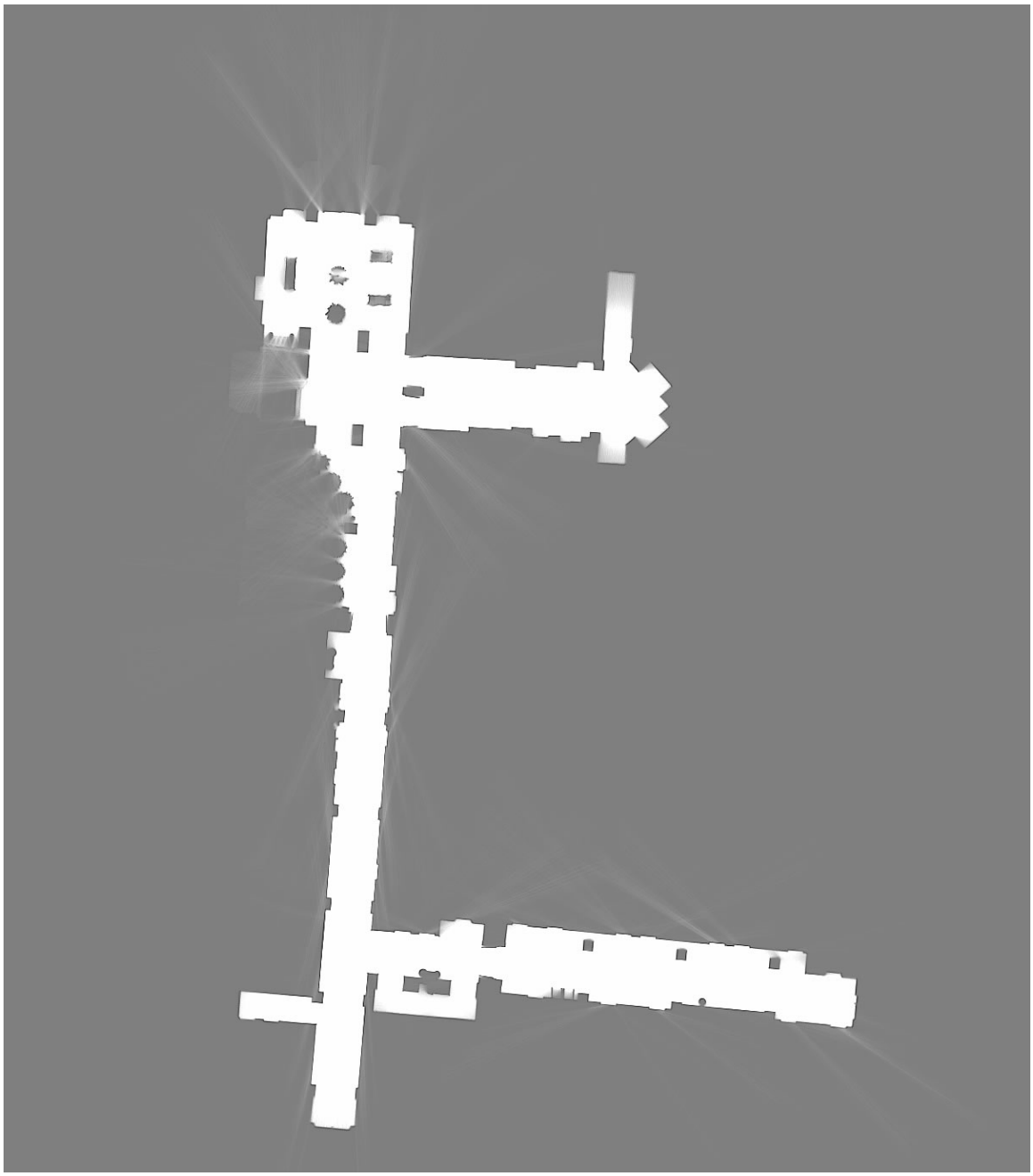

Figure 8: Laser map of Claxton Education 2nd floor created autonomously by a Pioneer robot using Player and Pmap.

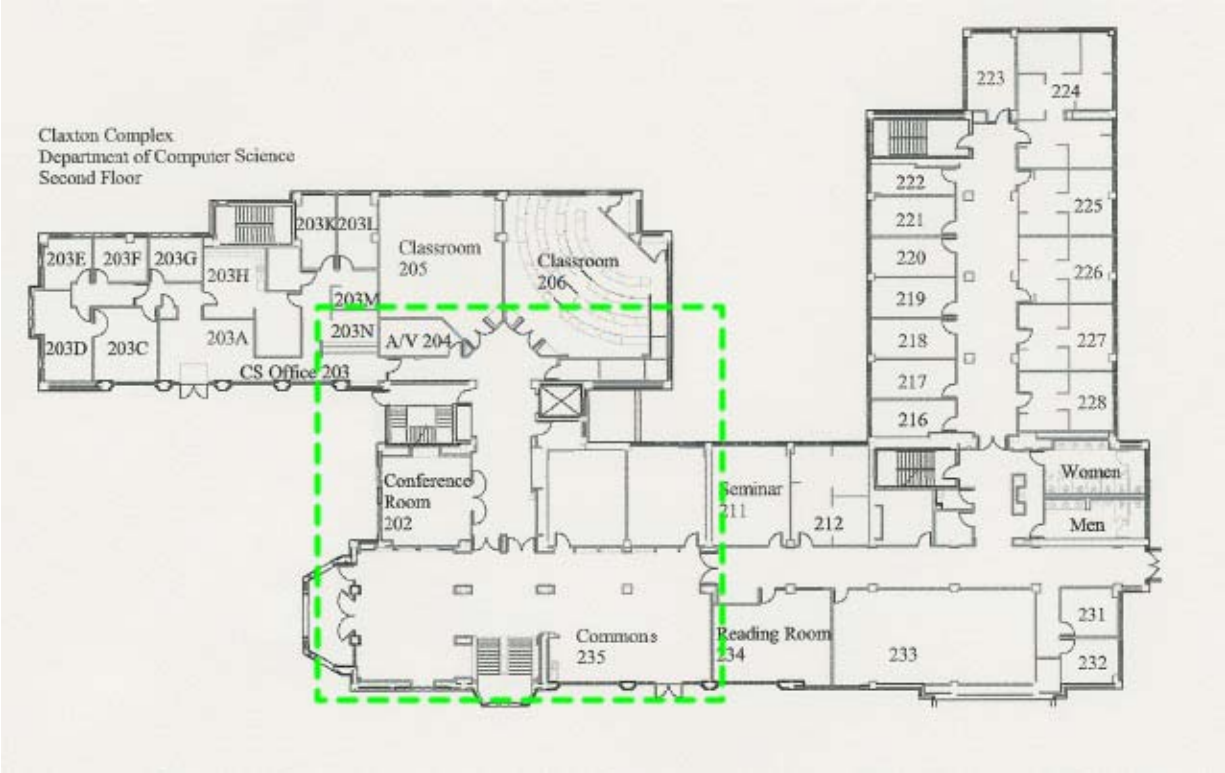

Figure 9: Building map of Claxton Education, 2nd Floor. Test area used in my experiments is outlined by the green dashed box.

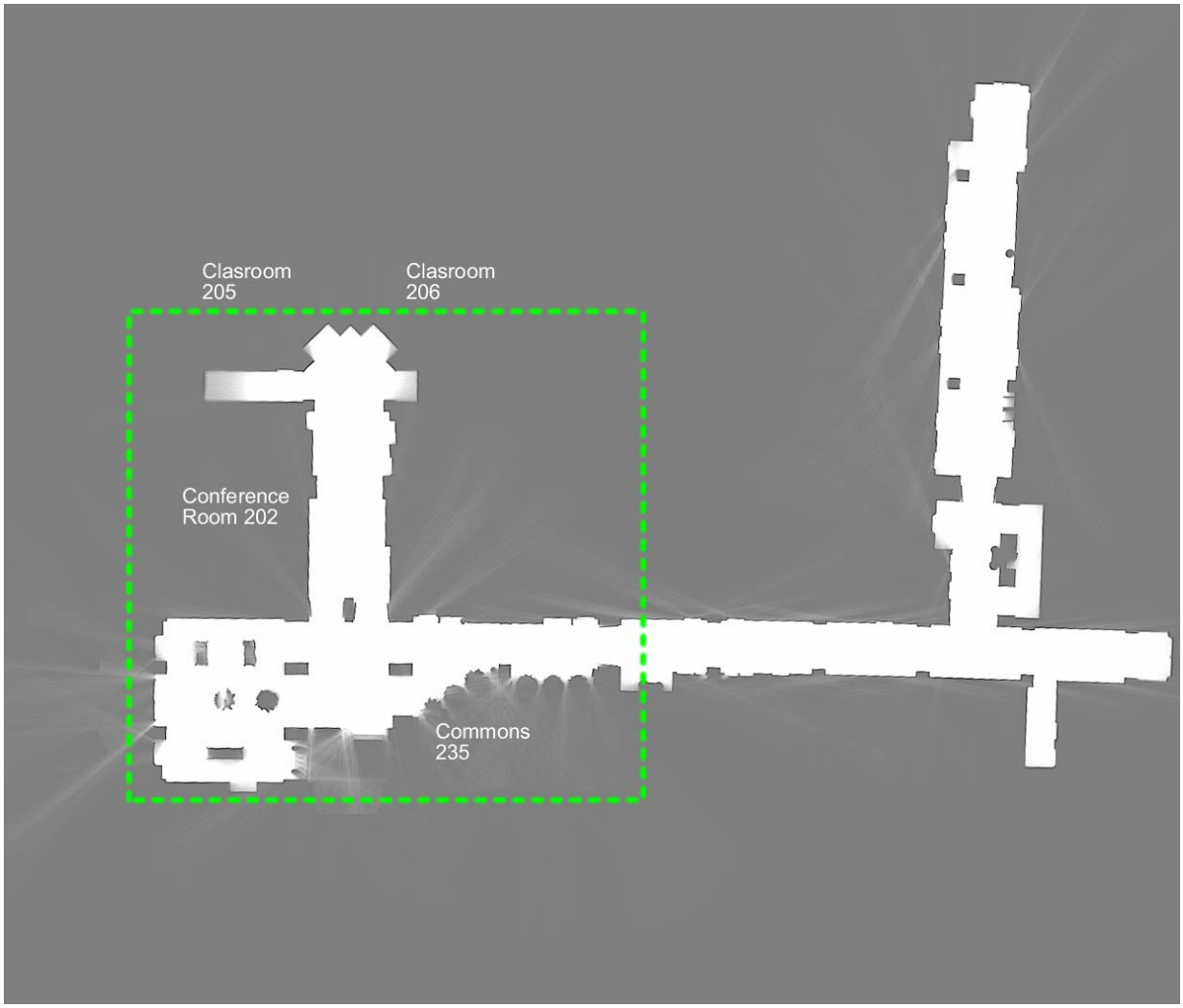

Figure 10: Autonomously generated laser map of area corresponding to Figure 9. The test area is outlined by the green dashed box.

and the target localization point. Because the correspondence problem is not a focus of this research, all localizations within the error margin (defined as 2m) of a target are treated as a correct localization, even if the target has been previously localized; all targets outside of 2m are considered incorrect.

3.2.4 UAV Hardware

This section covers the UAV hardware, its parts, construction and capabilities.

UAV Construction

Because of the delicate nature of lighter-than-air vehicles, including the need to miniaturize and conserve as much weight as possible while preserving stability and maneuverability, much experimentation went into the initial phase of the UAV construction. Originally, the hope was to purchase an off-the-shelf model blimp. However, the model blimps available for purchase are typically around 3' long, are relatively unstable and extremely susceptible to drafts, and only have sufficient lift for their gondolas and nothing more. Because the UAV must be reasonably stable in flight and because the desired payload included a camera and battery, a model blimp was deemed insufficient. Therefore, a custom approach was chosen for this project.

The initial design of the UAV was to have two fans, each individually controlled, on a rotating axle. This design is similar to the large remote-controlled commercial blimps popular at sporting events. The physical construction began with a 7' metalized nylon blimp envelope purchased online and off-the-shelf parts purchased from a local hobby store. The gondola frame was constructed out of balsa wood, cut to accommodate the hardware components. A large HS-785HB HiTech winch servo was chosen for its high degree of rotation to control the axle via a pulley, which allowed the axle to rotate just over 360°. Two EPS-300C GWS motor & gearbox assemblies (with propellers) were mounted on either end of the axle. Control over the motors was achieved with two ICS-300E GWS electronic speed controllers (ESCs). The ESCs available were one-way only, meaning the motors could only run in one direction. Power was supplied by a 9.6V 650mAh NiMH battery pack.

The initial UAV design provided several insights into the requirements and desired qualities and the tradeoffs between weight, power, and control. The initial design was too heavy; the lift provided barely supported the weight of the gondola, without a camera. The thrust generated by the motors was much greater than required, causing frequent oversteering and overshooting the destination. The over-powered motors, together with the need to rotate the axle to reverse thrust and adjust altitude, resulted in a general lack of control. The metalized nylon envelope, while sufficient for the task, was discovered to be very fragile and prone to "sag" when less-than-full. In addition, the balsa frame was very delicate and prone to crack around high-stress areas (e.g., the legs and the pulley).

Taking all of the lessons learned from the initial UAV construction, a new design was undertaken, focusing on minimizing the weight and sacrificing unneeded power. A new envelope made of polyurethane plastic was selected for its superior durability and semi-elastic nature, which reduces the "sag" problem when not fully inflated. Because the polyurethane is slightly lighter than the metalized nylon, a smaller 6' envelope was

also chosen. The tradeoff for this lighter, more durable envelope is that polyurethane is less of a helium barrier than the metalized nylon, which causes the blimp to deflate faster as helium slowly diffuses through the bag (but not fast enough to impact an experiment). The 6' envelope holds approximately 18 cu ft of gas and weighs 6 oz. So, at Knoxville's elevation, 936ft, the 6' polyurethane envelope filled with helium provides approximately 12 oz (net) lift.

A new, stronger and smaller gondola frame was constructed out of custom-machined aluminum plates with balsa wood supports. A "blimp kit" was found and purchased from a vendor that contained a specialized, lightweight dual 2-way proportional ESC and three micro motors. The dual two-way ESC allows two-directional (forward and backward) control of two channels. The new design mounts two of the micro motors on the axle, as before, but both are connected to one ESC channel and operate in unison. Horizontal rotation of the UAV is achieved, not through individual control over the left and right motors, but rather through the use of a third tail-mounted motor, which is connected to the second ESC channel. Altitude can be changed by rotating the axle and using the left and right motors. A micro servo rotates the axle and is geared up to produce over 180° of rotation. Because of the two-way ESC, 360° rotation was not necessary, so a much lighter "micro" servo was able to be used.

A standard, two stick, hobby-type 4-channel transmitter / receiver (Figure 11) was purchased and used for remote control. The transmitter and receiver operate one AM 27MHz Channel 6. After experimentation to determine the ideal controller configuration, forward/backward on the left stick was set to control the left/right thrust fans, forward/backward on the right stick rotates the axle, and left/right on the right stick yields left/right rotation via the tail fan. The tail motor was put on receiver channel 1, the micro servo on channel 2, and the thrust motors on channel 3. Channel 4 was unused.

Power for the UAV is provided by a custom-built 4.8V 160mAh NiMH battery that weighs only 17g and provides enough power to operate the UAV for about 20 minutes per charge. Such a small battery combined with a fast charger allows for the battery to be recharged in less than one hour between experiments. Three NiMH batteries were purchased to allow for several experiment runs to be conducted in sequence. Power for the camera is provided by a standard 9V battery.

The video camera chosen for this project is an Eyecam 2.4GHz Color Micro Wireless Video Camera. The camera contains an onboard transmitter and weighs less than 20g total. Care was taken to select a camera that transmitted on a channel (channel 9 - 2452 MHz) not used by the campus wifi network (see Figure 12). The camera system came with a camera mount and a receiver/composite video output device. Video output is displayed on an HP projector with composite input.

The newly designed UAV assembly is shown in Figures 13 and 14. The final weight for the complete assembly (including everything except the envelope) was 11.1 oz. Since the total lift was approximately 12oz, a small amount of ballast had to be added before flight to neutralize the buoyancy. Figure 15 shows the assembled complete UAV.

UAV Parts List

The following is a list of the parts used to construct the UAV.

Figure 11: The basic 4-channel 27MHz transmitter used to control the UAV.

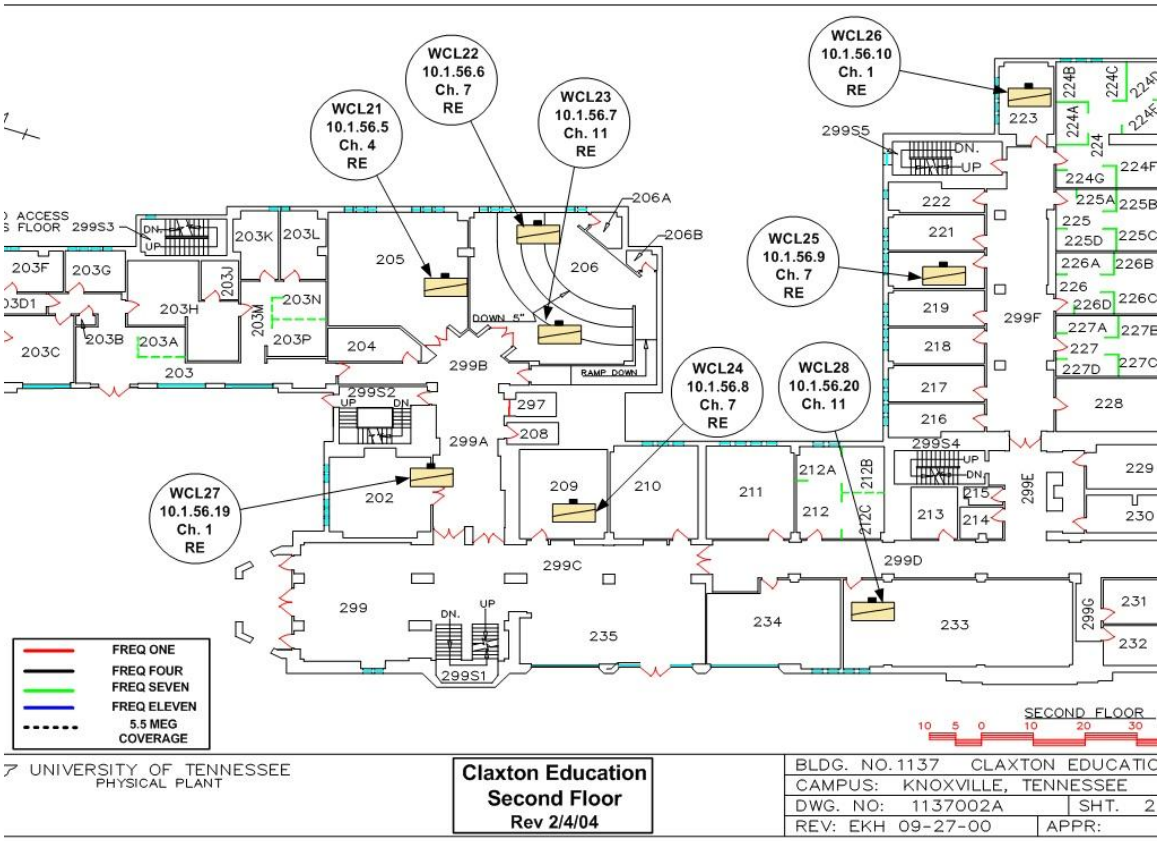

Figure 12: Access points (Channels), Claxton Education Building, 2nd Floor**.**

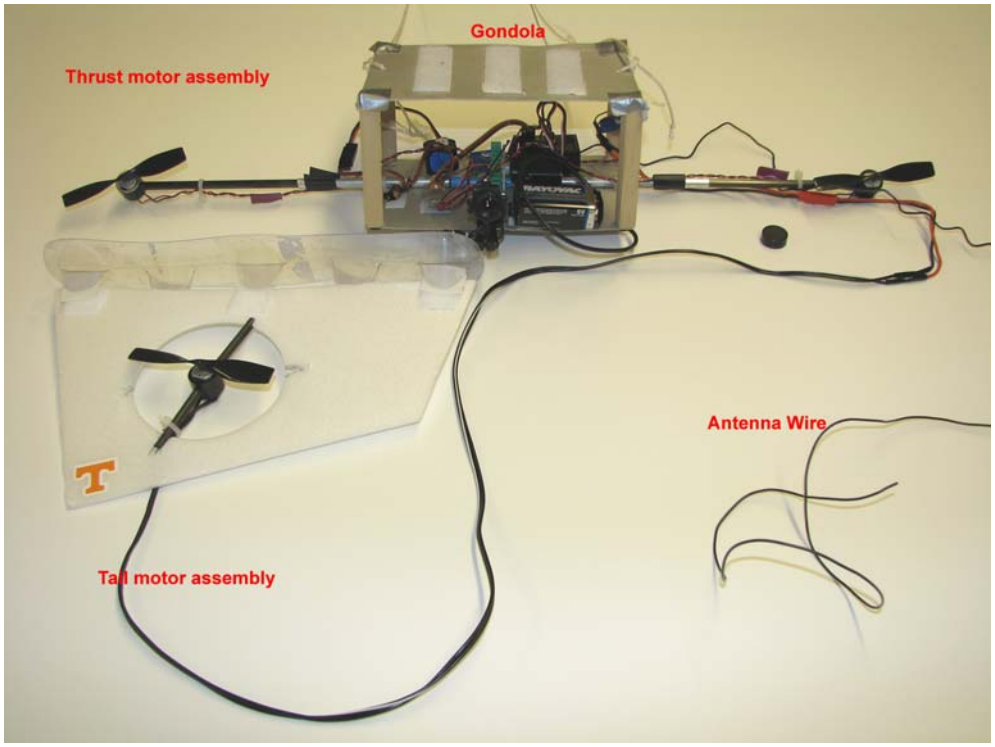

Figure 13: UAV motor assemblies and gondola.

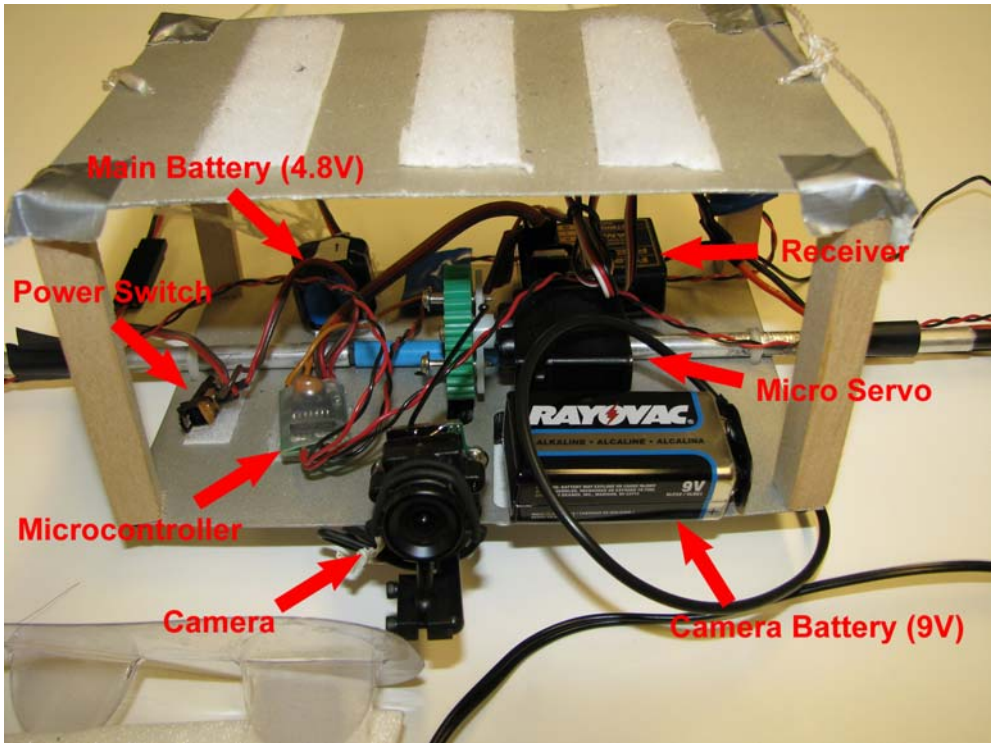

Figure 14: UAV gondola components.

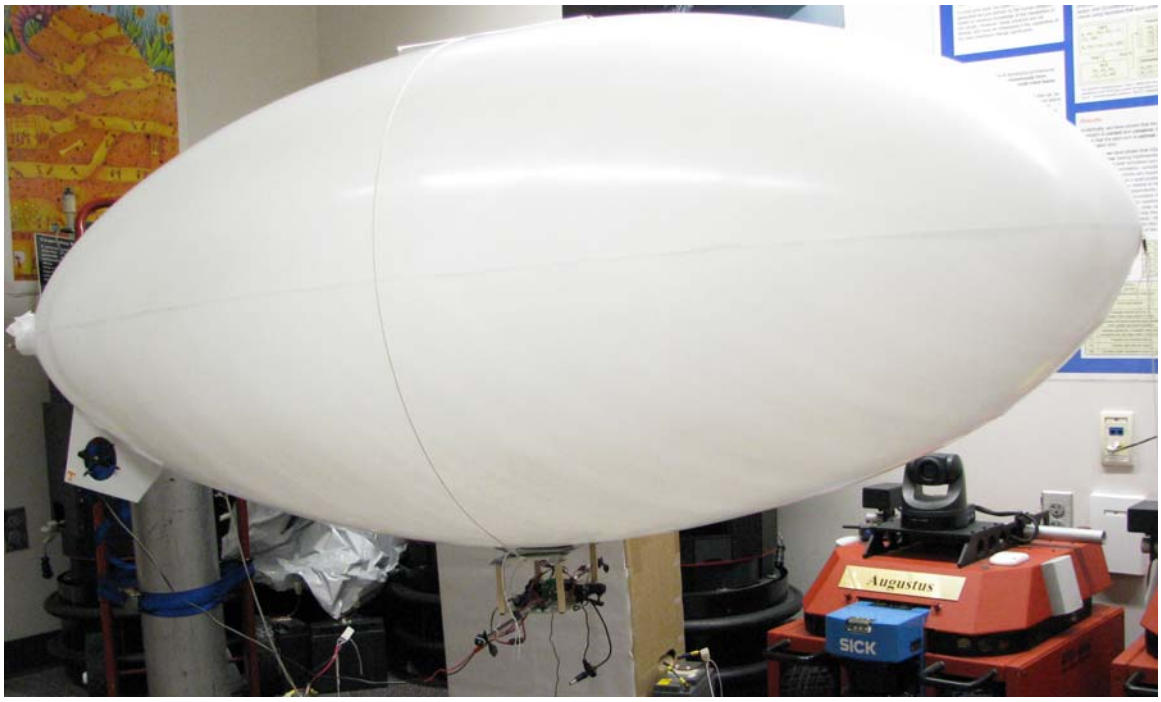

Figure 15: The Unmanned Aerial Vehicle, completely assembled, floats tethered in the lab.

- HS-81 HiTech Micro Servo
- Gear set, Stevens International No. MR7
- 4 channel receiver & transmitter, Futaba Attack Digital Proportional R/C System, AM27 MHz channel 6
- Balsa wood, 3/8" square
- Carbon fiber rod, 3/16"
- Aluminum tube, 1/4"
- Speed Controller - dual computerized proportional control [3]
- Fins and Fin Mounts
- Velcro
- 3x 2.5" Propellers
- 3x N20 Motors & Holders
- Eyecam 2.4GHz Color Micro Wireless Video Camera System - channel 9 (2452 MHz), 92° f.o.v., f=3.6mm [5]
- 6' polyurethane blimp bag 18cuft/6oz [1]
- Custom-ordered 4 x 160mah AAA cell 4.8v NiMH battery 0.6oz [4]
- 9V battery for camera

UAV capabilities

Control:

The UAV is teleoperated by a human pilot via wireless RF remote control. The range of the control is sufficient to reach all areas of the test environment from a central location.

Motion:

The two main motor/propeller assemblies are attached to a shaft parallel to the ground and perpendicular to the flight of the UAV. These motors provide thrust for the UAV's movement and are reversible (forward/backward). Because the shaft is rotatable a full 360°, thrust can be supplied in any combination of forward, backward, up, and down.

A third motor/propeller assembly located on the tail of the UAV provides rotation and is operated independently of the two thrust motors. This motor is also reversible, and can therefore rotate the UAV clockwise or counter-clockwise.

These three motors provide all of the movement capabilities of the UAV. It is worthwhile to note that these capabilities, while sufficient, are limited. Side-to-side movement is not possible, for example; instead, the UAV can rotate and move forward/backward.

The somewhat limited movement capabilities combine with several other factors to make piloting the UAV a challenging endeavor. Lightweight and streamlined, the UAV experiences no practical friction when moving unimpeded; once moving in a direction, it tends to stay moving in that direction until the motion is cancelled via the thrust motors or until it impacts a wall or obstacle. To illustrate the difficulty this presents to the inexperienced pilot, examine the simple task of turning a corner. Approaching a corner, the UAV is moving forward. At the right moment, the thrust fans must provide the correct amount of thrust to cancel all forward momentum. The tail motor must then be run briefly to provide enough force to rotate the UAV 90°. Then the tail motor must be run again to provide enough force to stop the UAV from rotating once it is facing the correct direction. Then, finally, forward force can be applied by the thrusting motors to begin moving in the new direction. All of this must be accomplished while simultaneously making minor adjustments to maintain the correct altitude. Since the UAV is especially susceptible to drafts and pressure changes in the environment, constant minor corrections must be made in all directions. Sideways drifting is especially problematic, since, as noted above, side-to-side movement is not possible.

Piloting:

Aspects of the environment also add significant difficulty to the task of piloting the UAV. Vents in the ceiling and walls can significantly alter the flight path, and intakes in the ceiling represent an inescapable vortex if the UAV gets too close to them. Temperature changes in the building after the sun sets also affect the buoyancy of the UAV.

Vision:

A color micro camera provides the UAV pilot with a limited, forward-facing view of the environment for both navigation and target identification/localization tasks (Figure 16). The field of view for the micro camera is 92°. Because the UAV can change altitude, both high and low targets can be identified.

Communication:

Figure 16: A screenshot from the UAV camera view of a target.

The piloting team uses a laptop connected to a private ad-hoc network via 802.11 wireless to communicate with the robot on behalf of the UAV. A diagram of the communication and control signals exchanged is shown in Figure 17.

3.2.5 Robot Hardware

This section covers the robot hardware, its configuration and capabilities.

Robot Configuration

The robot used for this experiment is a Pioneer 3 DX model (Figure 18) owned by the Distributed Intelligence Laboratory, part of the Department of Electrical Engineering and Computer Science at the University of Tennessee in Knoxville. The robot, named "Arno", features an onboard Pentium III computer running the Gentoo Linux operating system, a SICK laser range finder, Canon VC-C4 camera, and 802.11 wireless. The PTZ camera was zoomed out to the full ~50° field of view, panned to face exactly forward, and tilted down ~15° from the horizontal for the duration of the experiment.

The 802.11 wireless was configured to join the DILab's private ad-hoc network to communicate with the UAV's computer.

The housing of the SICK laser range finder was partially covered with white paper as the shade of blue paint on the housing would sometimes have an RGB value within specified tolerances of the targets' color and would be detected in reflections off of glass surfaces in the environment.

Robot capabilities/control

The robot runs the Player robot server [2][11] (ver. 1.6.5) which provides proxy interfaces for most aspects of the robot's hardware. Software was developed for this project that uses the Player server and gives the robot the ability to autonomously perform all of the behaviors and abilities required for this experiment. Those behaviors and abilities include movement, self-localization, navigation, communication, and target detection and localization.

Movement is controlled via the Player positionproxy interface. This interface allows the behavior program to issue direct, simple, movement commands as well as access basic position and orientation information.

Localization is accomplished by Player using odometric data combined with the SICK laser rangefinder and previously-generated map using a particle filter Monte Carlo localization approach. Localization information is used by the wavefront "pathplanner" proxy, which provides navigation information and control.

Communication is accomplished with simple network sockets.

Target detection and target localization are performed using data returned by the Player blobfinder proxy. The blobfinder proxy is provided RGB values of colors to look for when the Player server is started, in the form of a configuration file. During runtime, the blobfinder proxy object can be accessed to provide position and dimensions of all objects in the camera's field of view that match those initial RGB values. As stated earlier, testing revealed that "green" (RGB [0, 255, 0], with thresholds of 20:200, 50:220, 40:115) is the most easily identifiable but unique color for use as targets in the

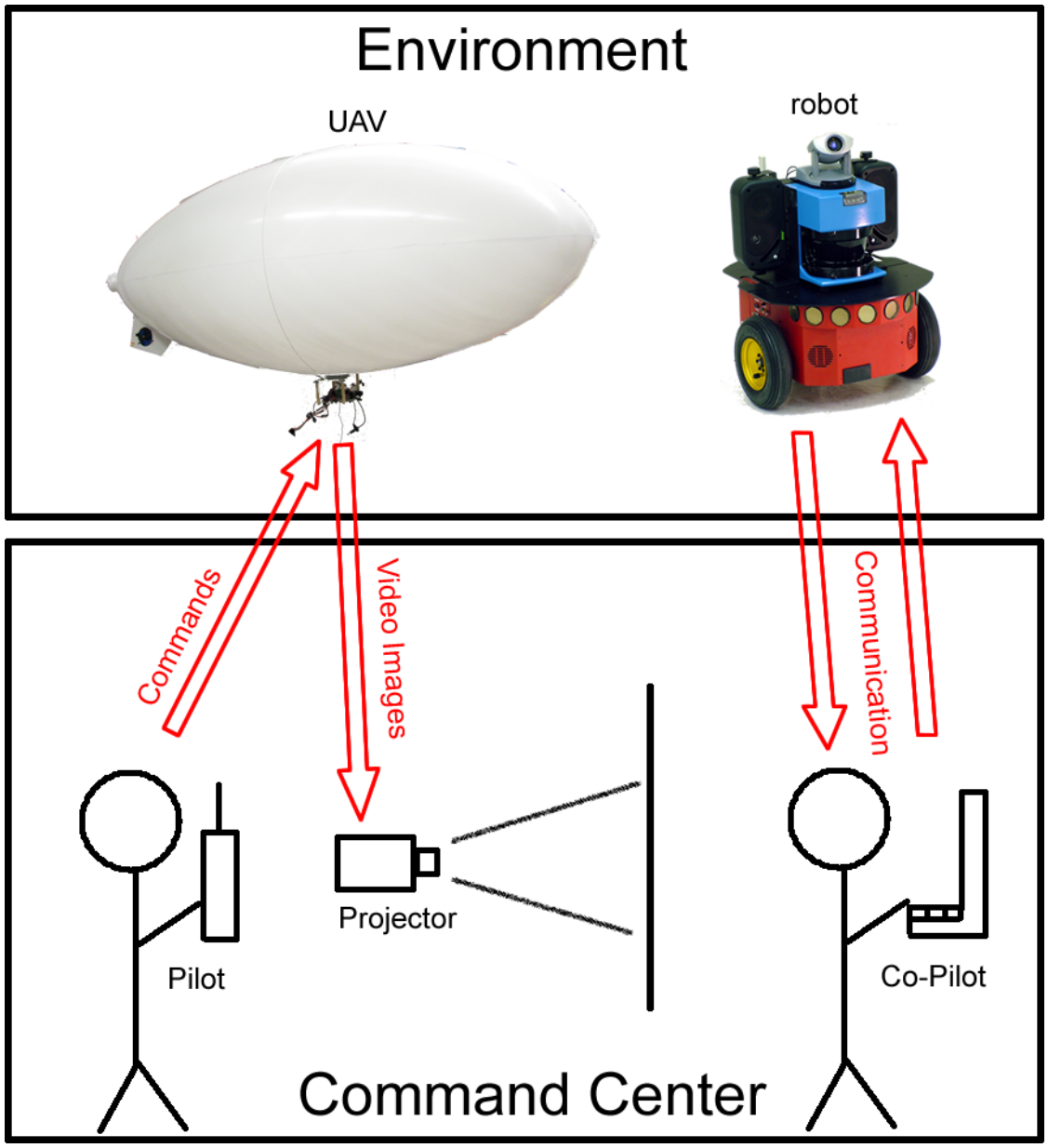

Figure 17: A basic illustration of the interacting elements involved in communication and control.

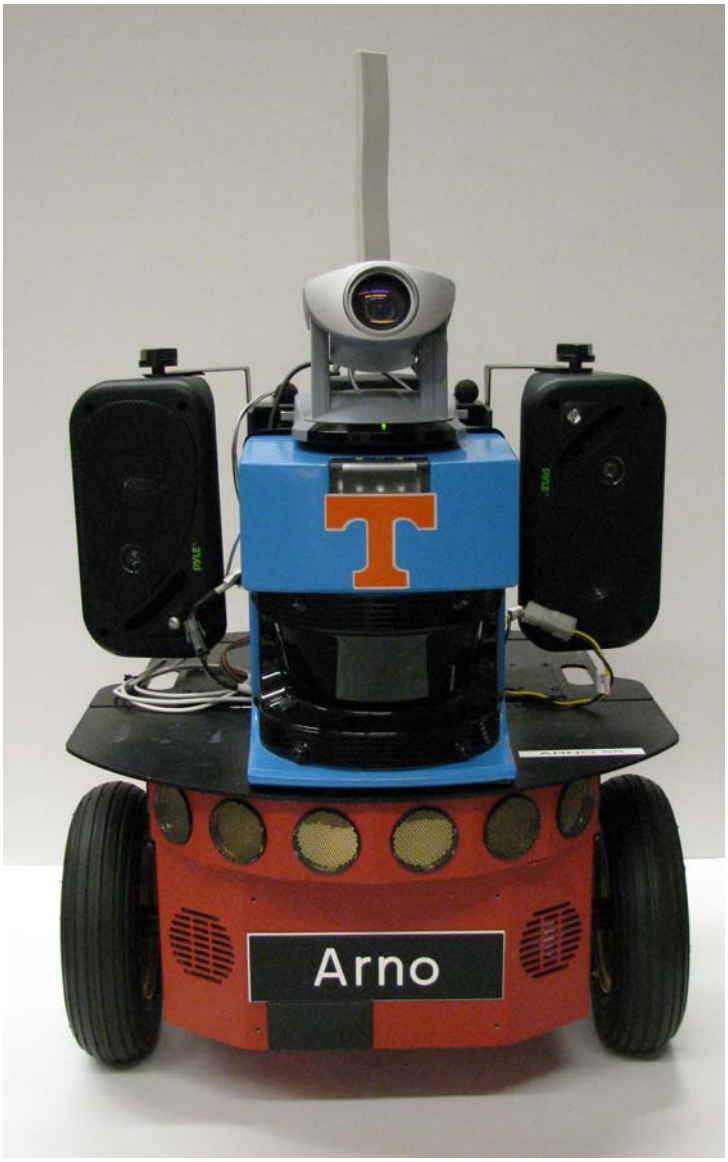

Figure 18: Arno, a Pioneer 3 DX robot.

environment. Relative dimensions of a tracked "blob" (Figure 19) are used to identify it as a target (the targets, balloons, are nearly spherical, so they should always have roughly the same, nearly equal, relative dimensions). Once a tracked blob has been identified as a possible target by its dimensions, blob size ("area") is also used to rule out blobs too small or large to be a target. In Figure 19, note the two small highlighted green rectangles in the top center of the photograph – the blobfinder reports all "blobs" within the specified color criteria, so it is necessary to discard blobs that are outside of an expected size range. Experimental testing revealed the accurate detection range to be 0.5-3.5m, and I implement threshold checks to limit detection to this range. If the size of the blob is within acceptable tolerances, blob size is then used to determine the approximate distance from the robot to the target, which, combined with relative position and current robot orientation, provides the position of the target.

Because the robot's camera is fixed in a slightly downward-facing position (about 15°), the robot is not capable of seeing "high" targets.

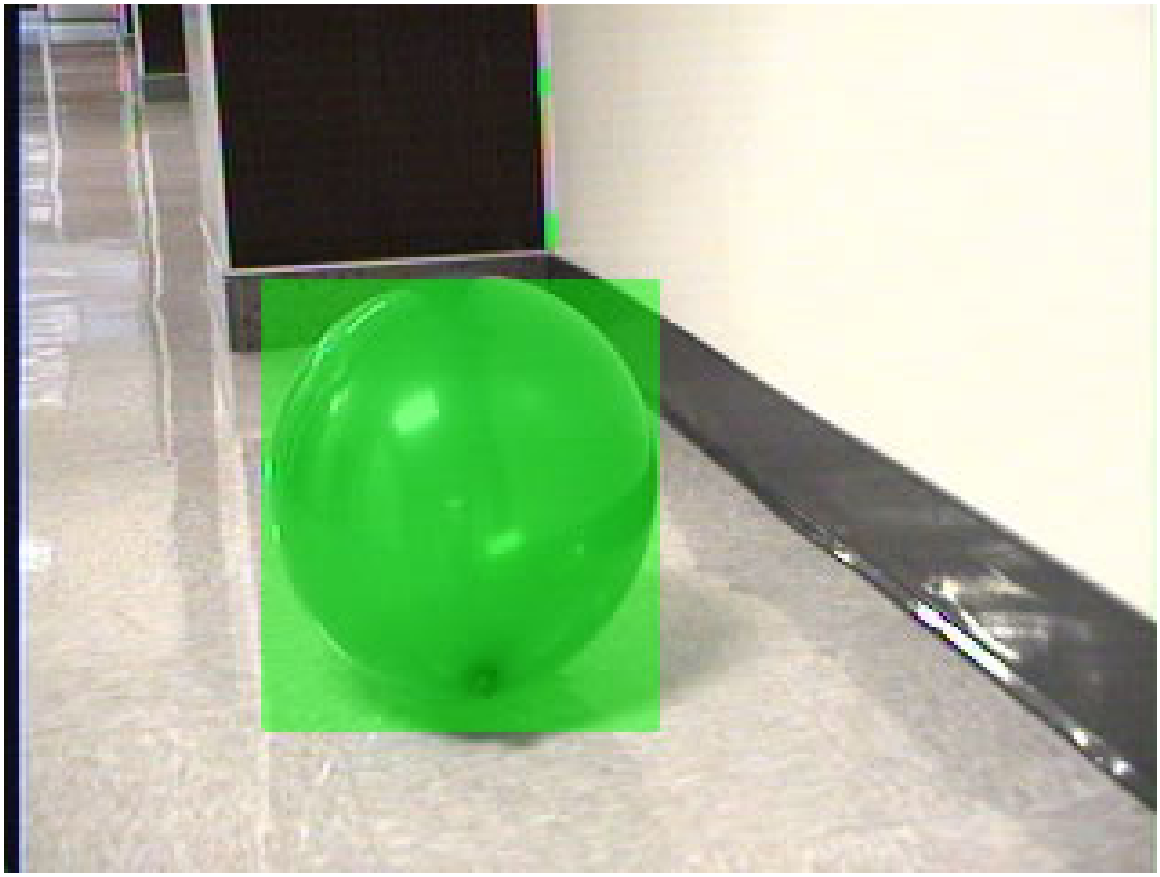

Figure 19: A target viewed from the robot's Canon VC-C4 camera taken via the Player "playercam" interface. Note that the green rectangular overlay over the balloon highlights the "blob" that the blobfinder reports.

4. Experiments

This section describes the exact experiments that were conducted, including setup, EDI definition, paths, experimenter roles, and robot behaviors.

4.1 Experiment Scenario

For the exact experiment scenario, the robot accomplishes the coverage/search task by moving through a pre-planned set of waypoints, beginning at the "entrance" to experiment environment near the fire doors south of the 2nd floor common area, and ending at the "exit" at the main doors in the 2nd floor lobby. At each waypoint, the robot rotates 360°, stopping every 36° to scan for a target, for a total of 10 scans. Even though the robot's camera has a 50° field of view, additional scans were found in experimental testing to add only approximately 3 seconds per scan, and this slight (7° per side) overlap greatly increases the detection rate.

Likewise, the UAV performs the task by navigating along a pre-planned path. The UAV is teleoperated by a human pilot in the "command center" situated in conference room 202, a location adjacent to but separate from the test environment (Figures 9 and 10). The pilot uses the onboard camera's feed to navigate. A human "copilot" assists the pilot in 1) target detection and localization via the camera feed and 2) communication with the robot on behalf of the UAV via software running on a computer in the command center.

The UAV piloting team has the ability to communicate three types of messages to the robot: 1) stop and await a command, 2) move to a specified set of coordinates and await a command, 3) proceed with normal functions ("go").

For the experiment, the "landmark" EDI types (Figure 1) identified were corners in the environment where two walls of substantial length (~2m or more) meet. These landmarks were chosen because they were found through testing to be easily identifiable, and because they are common but not overabundant in the environment. Figure 20 depicts the landmark-type EDIs and EDI ranges in the test environment. So, when the UAV is within 3m of these EDIs, the UAV can localize on its own.

4.2 Experiment Assistants

Three or four volunteers assisted with the experiments each day. All were graduate EECS students, with the exception of one business Ph.D. Roles performed during each experiment included UAV pilot, UAV communications/target detection, and an impartial "safety" person who ensured the safe operation of the UAV and robot in the environment. An additional volunteer conducted video recordings of some runs.

4.3 Experiment – Behaviors Tested

Three test scenarios are defined to compare the different behavioral approaches – independent, cooperative-reactive, and cooperative-planned.

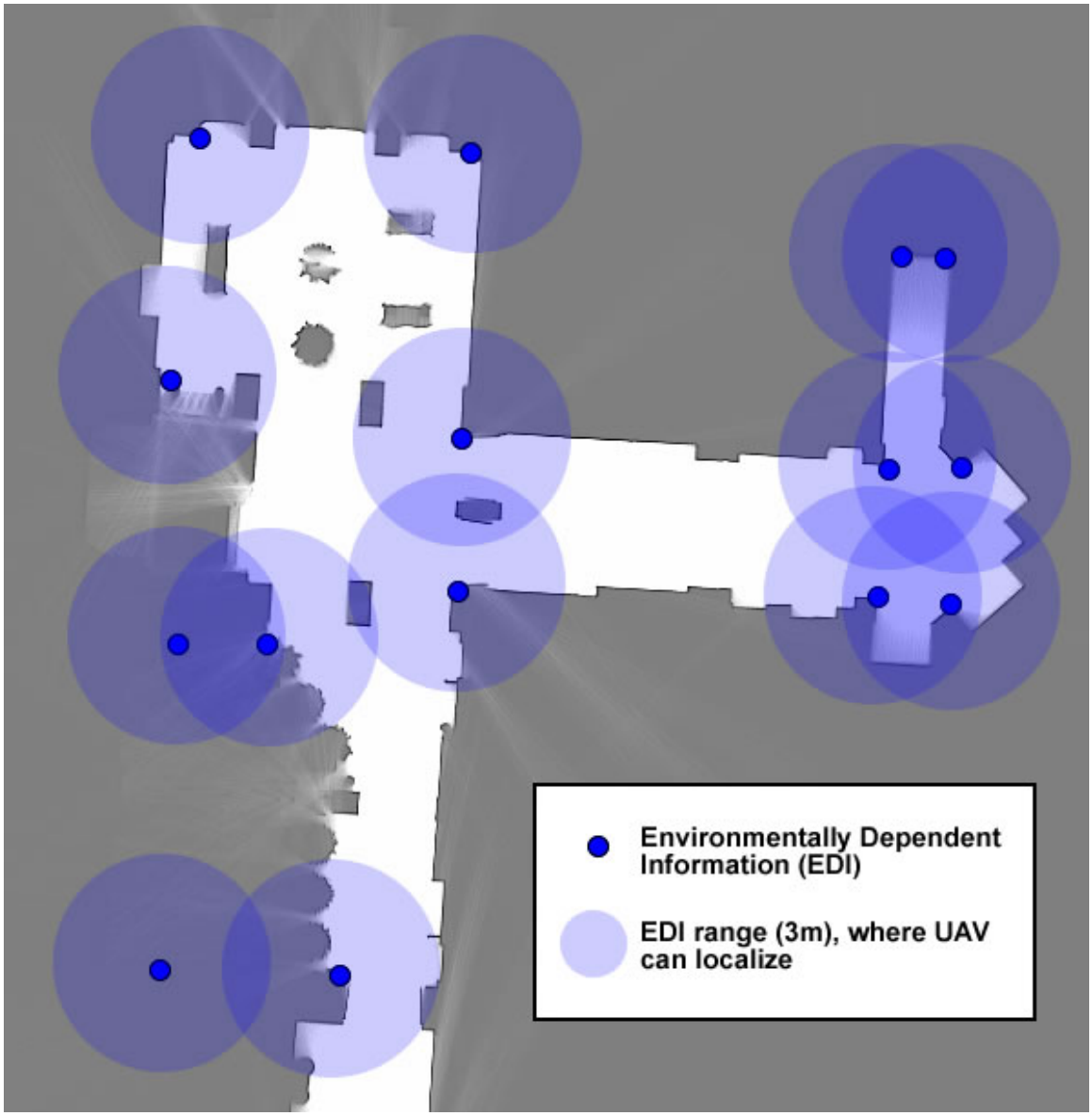

Figure 20: "Landmark" Environmentally Dependent Information (EDI) types in the test environment (small blue circles) and their ranges (larger light blue concentric circles). These landmark EDIs enable the human driving the UAV to localize.

4.3.1 Independent

In the independent test scenario, the UAV and the robot search the same target configuration "map" for targets, attempt to localize them, and operate independently of each other. This behavior scenario represents the simplest configuration for the team.

Note the differing capabilities of the robot and the UAV in this scenario: the robot, with its short height and fixed camera, cannot localize "high" targets. Comparatively, the UAV cannot localize any targets outside of the range of an EDI.

4.3.2 Cooperative Planned

In the Cooperative Planned scenario, the paths of the robot and UAV coincide at segments where cooperation is required. In the execution of the Cooperative Planned behavior, the robot proceeds through its pre-planned path.

Critical Juncture points in both the robot and UAV path are preplanned using ASyMTRe. As explained previously, Critical Junctures bound the regions of the UAV and robot paths where the robot and UAV should actively cooperate to increase performance. Figure 21 shows example paths of the UAV and robot (in dashed red and solid green) and the cooperative regions (boxed in black) overlaid on the EDI map from Figure 20. Thus, when navigating through such a cooperative region of their paths, the robot stops at each waypoint and communicates its position to the UAV, as well as its state as being ready to proceed to the next point at the UAV's request. The UAV then takes the opportunity to use the robot's communicated position to perform a relative localization of any nearby targets. After searching that area for targets, the UAV sends the command to proceed to the robot. When navigating outside of a cooperative region, the robot and UAV proceed searching along their paths independently. If either the robot or the UAV reach a CJ that denotes the beginning of a cooperative region before the other, they wait for the other team member. Also note that neither will stop and wait if a waypoint is not within a cooperative region.

This cooperative behavior allows the UAV to localize all targets in the environment. However, since the robot's scanning is fairly time-intensive, the UAV is usually waiting for the robot, which means that the UAV's runtime should be very close to the robot's runtime.

4.3.3 Cooperative Reactive

To execute the Cooperative Reactive behavior, the robot and UAV proceed exactly as they would in the independent scenario, with one major exception. As the UAV identifies and attempts to localize targets, if ASyMTRe reveals that the UAV lacks the information necessary to operate the target localization schema (i.e., the UAV is outside of the range of an EDI), ASyMTRe generates an alternative task solution where the robot assists the UAV in localization. Then, the UAV operators communicate a message to the robot to proceed to a position near the UAV and the detected target. Upon receipt of this position command, the robot decrements its current waypoint value (conceptually pushing its current goal onto its path "stack") and proceeds to the position requested by the UAV. Once there, the robot communicates its presence and exact

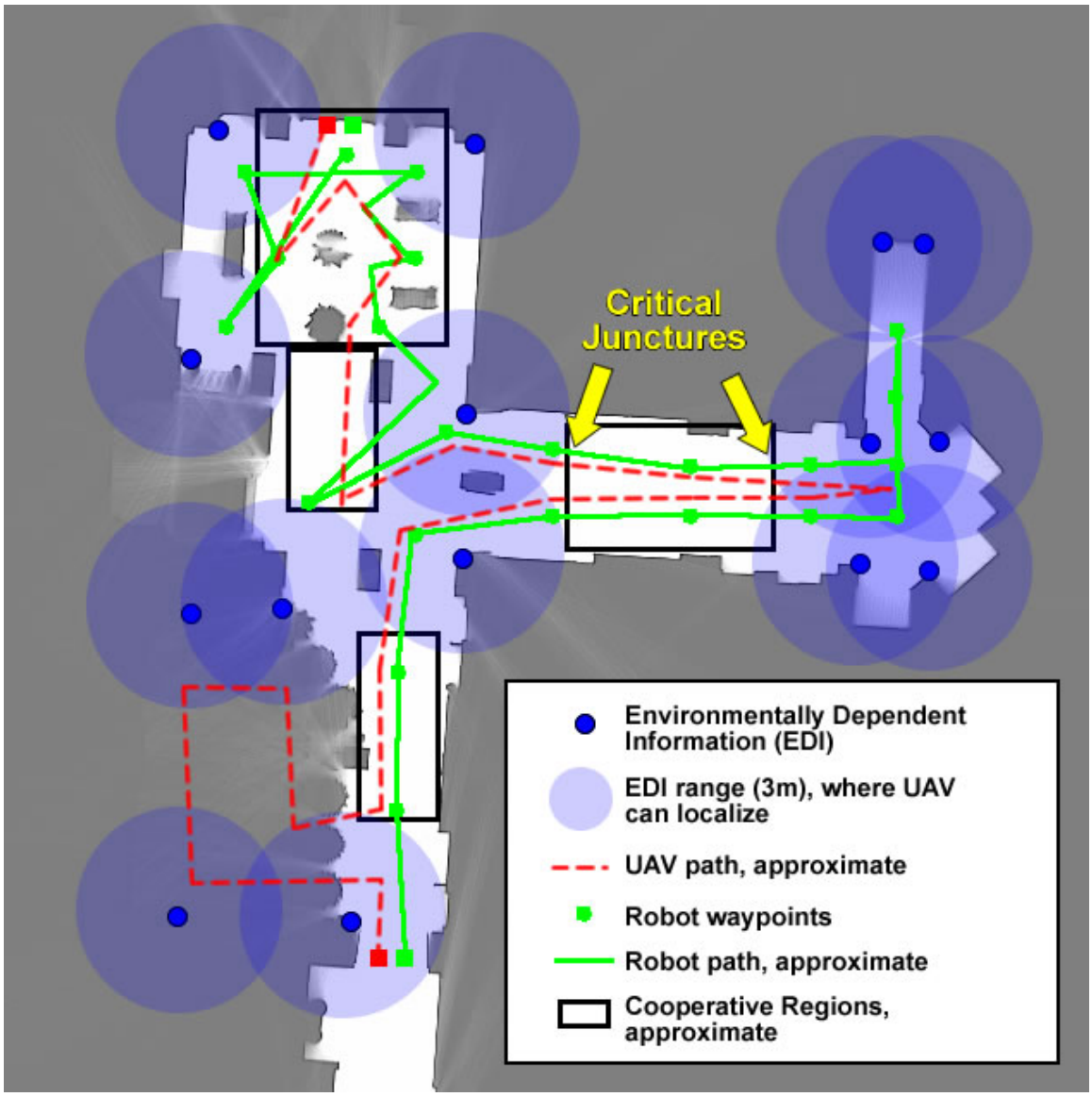

Figure 21: Cooperative regions. Dashed red line indicates the approximate UAV path, green line indicates the approximate robot path, black boxes encompass approximate cooperative regions, blue dots represent EDIs, and large blue transparent circles represent EDI ranges. Two approximate critical juncture locations are indicated by yellow arrows.

position to the UAV. Then, the UAV is able to visually localize the target relative to the robot. At this point, the UAV communicates a message to resume previous behavior to the robot, and the robot loads its last position as a goal position and returns to the independent search behavior.

The premise behind this cooperative reactive behavior is to be more robust than the planned cooperative in terms of not requiring preplanning and foreknowledge, and to streamline the cooperation behavior to only CJs where targets are detected.

4.4 Experiments Conducted

Five maps containing twelve targets each were created (Figures 3-7). Target positions were generated randomly and subjected to the following constraints: eligible positions were 0.5 m apart (i.e., the granularity of the position placement was 0.5 m); targets must have a minimum of 4 m between them; targets could be high (above 1 m) or low (on the floor) in the environment, with the exception of some spaces where only "high" was available (e.g., tabletops); all open space in the test area was eligible with the exception of some areas in the middle of hallways that were excluded beforehand for safety (i.e., so the robot wouldn't run over the target).

Experiments were conducted from 9 June 2008 to 17 June 2008. Because the Claxton Education building is often inhabited, even in summer, all experimentation was done at night and on weekends to avoid interference as much as possible and to minimize the impact on educational activities in the building. Occasional false starts due to human experimenter error or outside interference occurred and were discarded. No other data was discarded. Twice the experiment in progress had to be paused to allow a class to exit the building. Both times the paused experiments were resumed successfully and the experiment run-time was adjusted in post-data collection to remove the period of the pause.

Maps 1 through 5 were tested in order. With 5 maps and 3 scenarios, a total of 15 runs (robot and UAV operating simultaneously) were conducted. Each run lasted approximately 1 to 1.25 hours, including setup time. As the environment took time and assistance to configure, as many runs as possible were conducted consecutively each session.

Freshly charged batteries were used for the robot and UAV for each experiment. Batteries available largely dictated the length of each day's experimentation session which was 4-5 hours, on average.

Images of a Cooperative Planned experiment in progress are shown in Figures 22 through 26. Pseudocode for the robot behaviors is available in Appendix A-3.

Figure 22: The UAV and the robot begin a run.

Figure 23: The UAV and the robot navigate the environment.

Figure 24: The UAV waits on the robot for localization assistance.

Figure 25: The robot assists the UAV in localizing a target.

Figure 26: Localization is complete; the UAV and robot continue.

5. Evaluation

To evaluate the performance of the behavioral approaches, I define four metrics: Target Localization Accuracy, Task Completion Time, Aggregate Run Time, and False Positive Rate. Target Localization Accuracy is defined as the ratio of number of targets localized to number of actual targets. Task Completion Time is defined as the maximum of the robot and UAV run times for a single run. Aggregate Run Time is defined as the sum of the run time of the robot and UAV for a single run. False Positive Rate is defined as the ratio of the total number of localizations reported to the number of incorrect localizations.

5.1 Target Localization

First, Target Localization Accuracy for each behavior is analyzed. Figure 27 is an example of the target localization results. Table 1 shows the ratio of targets localized to actual targets in the environment, where 1.0 represents all targets found. In order to have a score of 100% localization possible, note that targets in the center of the Commons area (lower right of test area in Figures 9 and 10, lower left area in Figures 20 and 21) that are outside the range of the EDIs in that region were counted as inside the range of an EDI (i.e., localizable). Maps 2, 3, and 4 each had one such target (Figures 4 – 6).

The most apparent observation from Table 1 and Figures 28 and 29 is that the Cooperative behaviors displayed the nearly same ability to localize targets, while the Independent was less capable. To determine the significance of this difference between the Cooperative behaviors and the Independent behaviors, I applied a Student's T-test to the target localization behaviors' average results from Table 1, comparing the Independent approach against each of the Cooperative approaches, and the Cooperative approaches against each other. The test confirms that the differences between the Independent / Cooperative Reactive and Independent / Cooperative Planned behavioral approaches are statistically significant, with a confidence level of 95% and 97.5%, respectively. There was no significant difference between Cooperative approaches.

Both of the Cooperative approaches performed better as a system: they localized a significantly greater percentage of targets than the Independent approach. The reason for this becomes clear when one considers that in any application there could be certain subsets of the overall task that can only be accomplished via cooperation. In this application, if there are any targets in the randomly generated map that can only be

Table 1: Target Localization Accuracy: Ratio of Targets Localized to Actual Targets.

	Map 1	Map 2	Map 3	Map 4	Map 5	**Mean**	**Std. Dev.**
Independent	1.0	0.83	0.92	0.92	0.75	**0.88**	**0.10**
Cooperative-Reactive	1.0	1.0	1.0	0.92	1.0	**0.98**	**0.04**
Cooperative-Planned	1.0	1.0	1.0	1.0	1.0	**1.0**	**0.00**

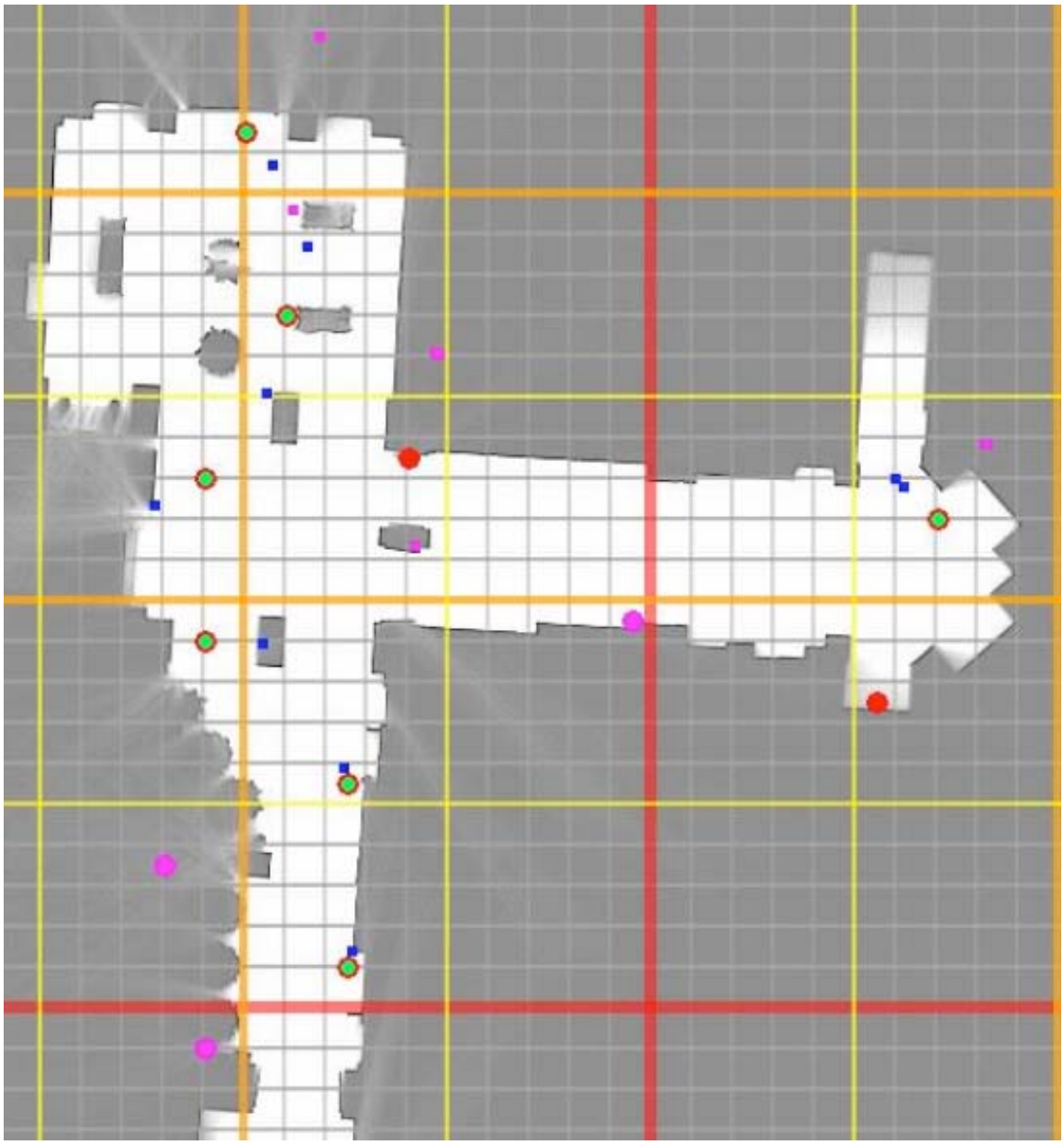

Figure 27: Localization results example for the robot on Map 4 using the Cooperative Planned behavior. In this example, the robot localized 7 out of 9 low targets. In the map, the following are represented:
Red dots – low targets.
Purple dogs – high targets
Red dots with green concentric dots inside – targets considered localized
Blue squares – localizations considered correct
Purple squares – localizations considered incorrect

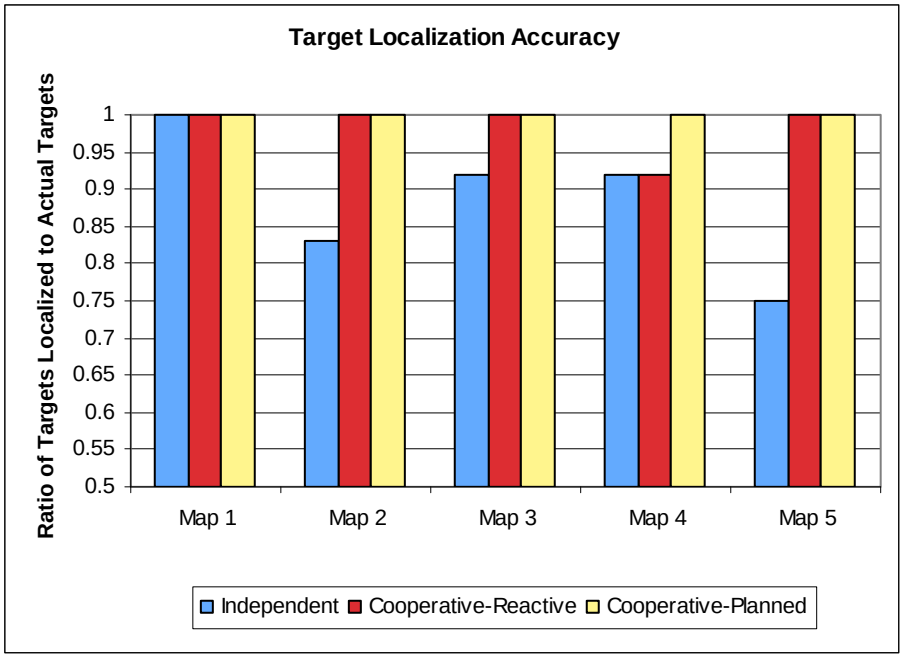

Figure 28: Target Localization Accuracy

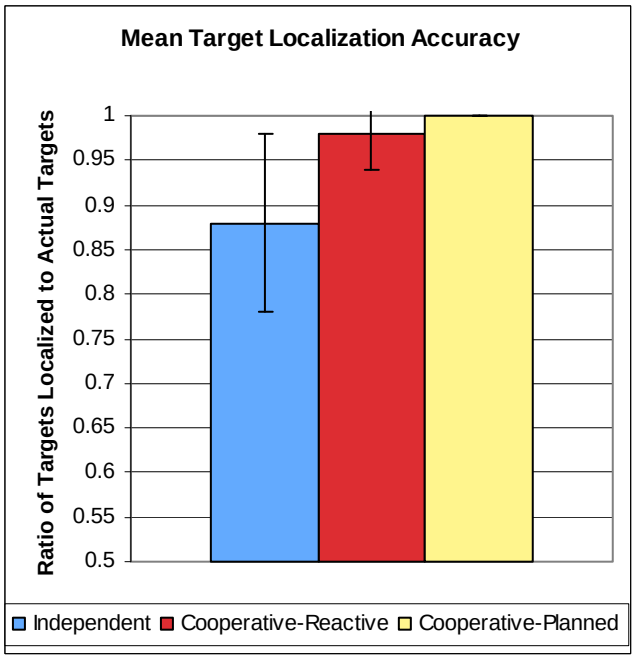

Figure 29: Mean Target Localization Accuracy, with standard deviations shown.

Table 2: Task Completion Time: Max of robot and UAV time per run, in minutes.

	Map 1	Map 2	Map 3	Map 4	Map 5	**Mean**	**Std. Dev.**
Independent	38.00	38.68	38.38	38.42	37.15	**38.13**	**0.60**
Cooperative-Reactive	43.43	43.93	43.00	46.15	48.90	**45.08**	**2.46**
Cooperative-Planned	40.17	40.82	39.69	39.22	41.03	**40.18**	**0.75**

localizable by the team in cooperation, then the Independent approach is unable to localize them. Thus, in this experiment, localization of these targets is representative of the larger issue of tasks that can only be accomplished through cooperation. So, in the general case, the performance of the Independent approach is bounded by the number of tasks that can be accomplished independently. In this case this is total number of targets minus those targets that are located high and outside the range of an EDI, which renders those targets viewable by only the UAV but not localizable.

5.2 Task Completion and Aggregate System Run Times

The next performance measures examined are Task Completion Time and Aggregate System Run Time. Task Completion Time values, the maximum time of the robot and UAV run times per run, are shown in Table 2 and Figures 30 and 31.

I performed a Student's T-test on the average in each pairing of rows in Table 2. Each pairing (Independent vs. Cooperative-Reactive, Independent vs. Cooperative-Planned, and Cooperative-Reactive vs. Cooperative-Planned) confirms a statistically significant difference in the Task Completion Time for each of the behaviors' approaches with a confidence of 99.5%.

The Task Completion Time of the robot and UAV in each run is equal to the run time of the robot, as the UAV is much faster than the robot because of its means of propulsion and the fact that it was human-operated. This is a measure worth noting because it represents the total time required by the team to accomplish the overall task. From these results one can see that there was not a huge difference in time between each approach. The Independent approach is of course fastest, as the robot has only to navigate and search along its path without any other interaction or interruption. Interestingly, the Cooperative-Planned approach is only slightly (~5% on average) longer – this can be attributed to the fact that, because the UAV was significantly faster than the robot, the UAV is always at each CJ at the same time as the robot. So, the UAV only has to take a few seconds to localize any nearby targets before issuing a "resume" command to the robot, thus impacting the robot run time a small, but statistically significant, amount. And of course the Cooperative Reactive approach takes the longest (about 18% longer than Independent), as during each run there are at least a few targets that the UAV requires assistance localizing, which sometimes requires a long round-trip for the robot from its search path to wherever the UAV requests assistance. It is worth noting that the

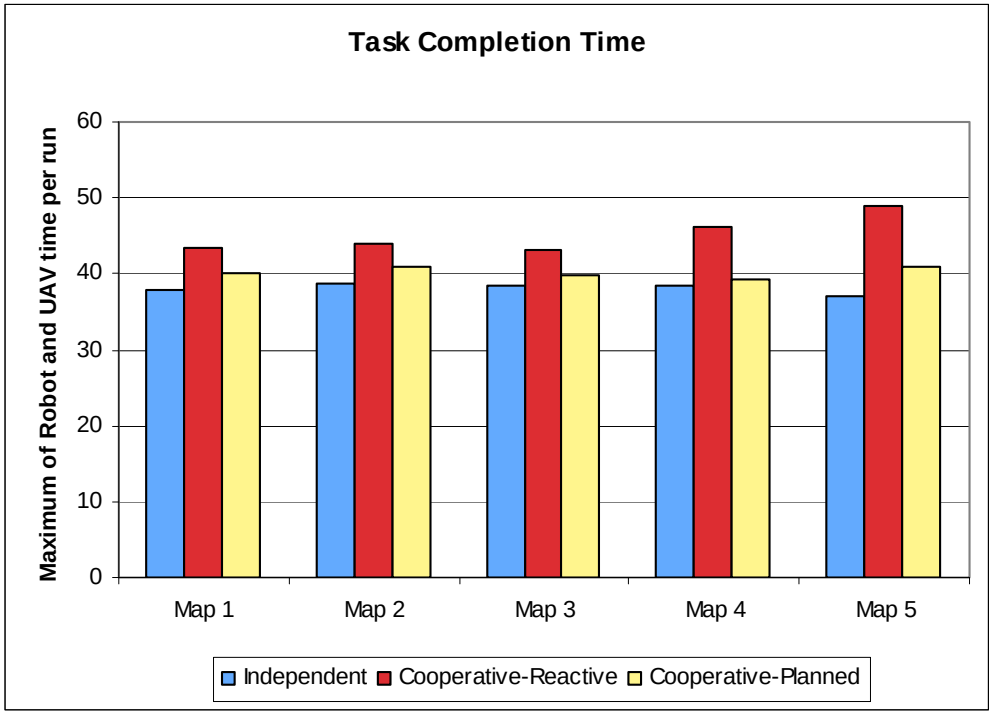

Figure 30: Task Completion Time

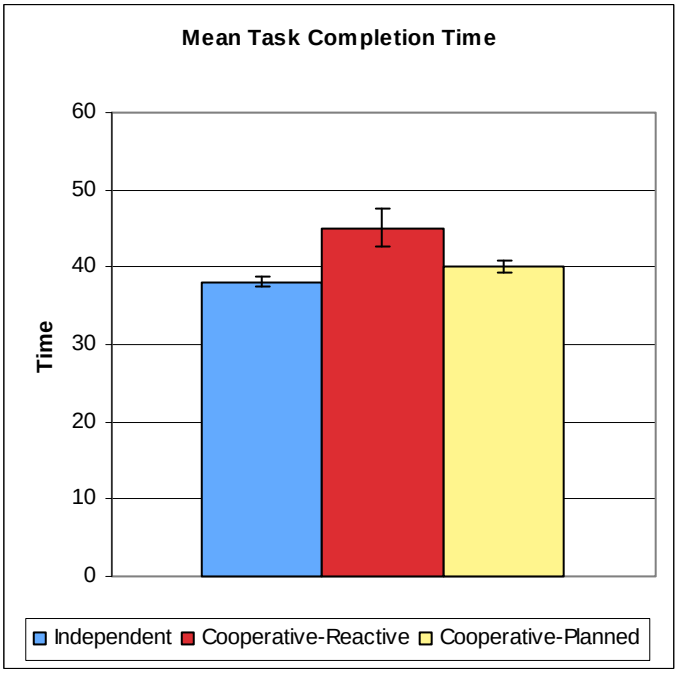

Figure 31: Mean Task Completion Time, with standard deviations shown.

Table 3: Aggregate System Run Time: Sum of robot and UAV time per run, in minutes.

	Map 1	Map 2	Map 3	Map 4	Map 5	**Mean**	**Std. Dev.**
Independent	46.00	46.18	47.38	48.42	44.15	**46.43**	**1.61**
Cooperative-Reactive	56.80	57.93	58.00	60.70	64.40	**59.57**	**3.06**
Cooperative-Planned	80.34	81.64	79.34	78.44	82.06	**80.36**	**1.52**

longer Task Completion Time of the Cooperative Reactive approach is directly related to the maximum speed of the standard Pioneer robot; with a much faster robot, the Task Completion Time of Cooperative Reactive should approach that of Cooperative Planned.

Aggregate System Run Time is the robot run time added to the UAV run time for each run. The Aggregate System Run Time results are shown in Table 3 and Figures 32 and 33. This value is interesting because it can be used as a very rough approximation of energy consumption of each team, as one can observe that at any given moment in experimentation, both team members are expending energy. The robot is nearly constantly in motion, and while one might think that the UAV would be idle while waiting for the robot in the Cooperative scenarios, the reality of blimp navigation is that in any real-world environment, there is a constant need to expend a non-negligible amount of energy simply to remain in a stationary position, as drafts and temperature changes are constantly affecting the UAV's position.

A Student's T-test of the data in Table 3 reveals that each pairing of average results is statistically different, with a confidence level above 99.95%.

Of course, the Independent value represents the sum of the times that it takes each team member to navigate and search its respective path. The UAV, as noted, is much faster than the robot at doing so. The Cooperative-Planned approach is a remarkably higher value than the other approaches (nearly 75% more than the Independent). This is because the Cooperative Planned approach involves the UAV and robot navigating their paths in synchronization, arriving at each CJ at the same time, since the CJs are generated a priori without knowledge of the targets' actual locations. Because of this, the much faster UAV must wait on the robot at each point, and therefore the UAV's run time is much longer in the Cooperative-Planned scenario. The Cooperative-Reactive approach is around 28% longer on average than the Independent, but much less than the Cooperative-Planned. This is a because the CJs are generated reactively as the UAV encounters a target outside of the EDI range, so while the UAV does have to wait for the robot to arrive and assist at some points, it does not have as many points at which to wait.

5.3 False Positive Rate

Because I allowed the robot to report redundant target localizations, a target could be localized more than once, as described earlier. The ratio of the number of the number of "incorrect" localizations (that is, those not within the error threshold (2m) of a target)

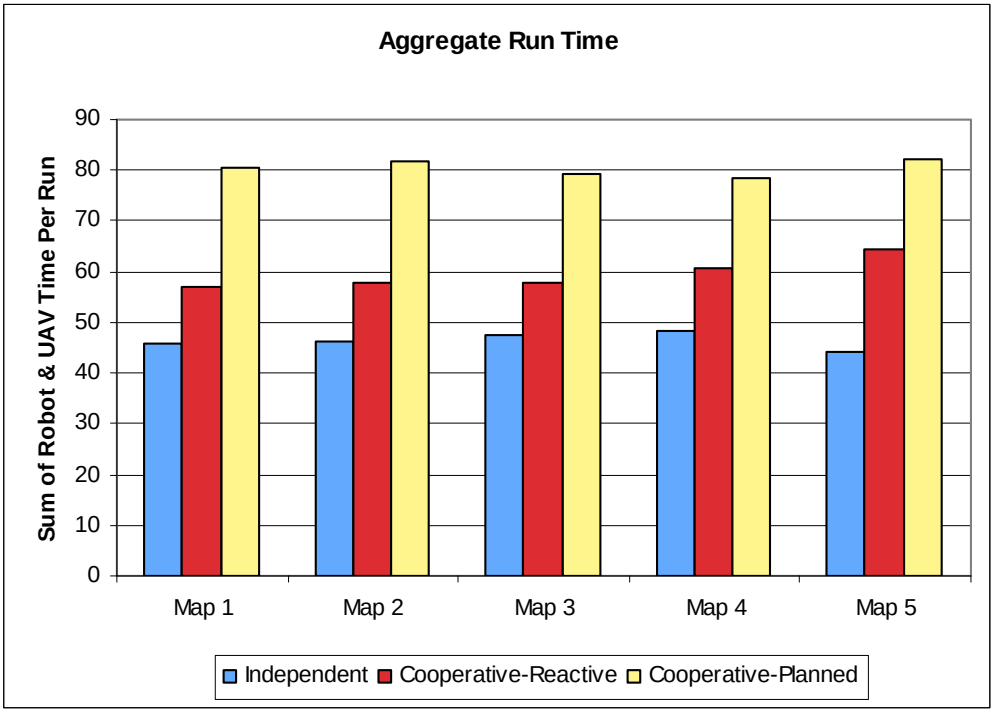

Figure 32: Aggregate System Run Time

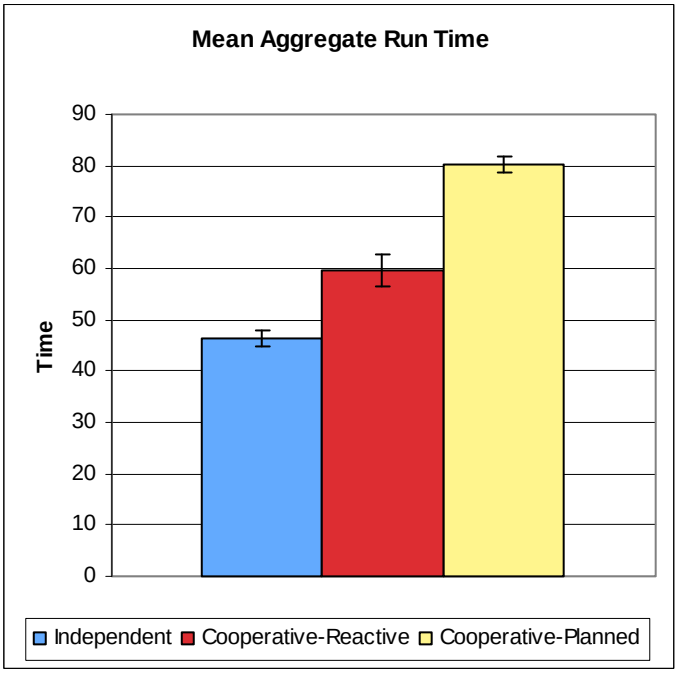

Figure 33: Mean Aggregate System Run Time, with standard deviations shown.

Table 4: False Positive Rate – 1 - Ratio of incorrect localizations to reported localizations.

	Map 1	Map 2	Map 3	Map 4	Map 5	**Mean**	**Std. Dev.**
Independent	0.28	0.37	0.28	0.20	0.33	**0.29**	**0.07**
Cooperative-Reactive	0.28	0.50	0.36	0.47	0.25	**0.37**	**0.11**
Cooperative-Planned	0.08	0.09	0.27	0.44	0.25	**0.20**	**0.15**

to reported localizations yields the ratios in Table 4, which I define as the False Positive Rate. Figures 34 and 35 depict the False Positive Rate and Mean False Positive Rate graphically.

A Student's T-test applied to the average of each row in the three pairings of data sets yielded no significant differences, except for the comparison between Cooperative-Reactive and Cooperative-Planned. These two data sets were found to be different to nearly a statistically significant amount, with a confidence above 90% and just less than 95%. It is my prediction that upon expansion of these data sets after further experimentation these data sets will diverge to a statistically significant amount and thus show that they are, in fact, different, and therefore that the Reactive approach yields more "false positives" than the Planned.

This performance difference could be reasonably attributed to slightly increased error temporarily introduced by the need for the robot to divert from the current path to assist the UAV and then return to its last position. First, it is necessary to note that the meaning of "false positives" here incorporates all target localizations that do not correspond to actual target positions: both total misses (e.g., a square patch of grass outside the lobby doors, or a green hue cast by an unusual lighting situation) and localization errors of actual targets in excess of the error threshold (e.g., due to orientation or distance errors). Examining the false positives, visually represented by the purple squares connected to purple circles (robot positions) in Figures 36-38, one can see that false positives are most often due to orientation errors and not distance errors or total misses. Note that at the maximum detection distance (3.5m), an orientation error (which yields an incorrect bearing-to-target measurement) of only about 32° can result in a 2m localization error. Coupled with the inherent error in the real-world system (slight odometry error, slight variance in size of target, environment lighting, background colors affecting target coloring, and so forth) a slight odometry error can yield occasional detections of actual targets that are simply outside of the tolerance range.

So, by sending the robot on long diversions from the path followed exactly by the Independent and Cooperative-Planned approaches, an additional, small but significant error must be being temporarily introduced into the Cooperative-Reactive system (Figure 38). This error would be best described as an error due to the "unsettled" state of the robot, as the orientation error appears to "settle out" eventually and return to normal; the

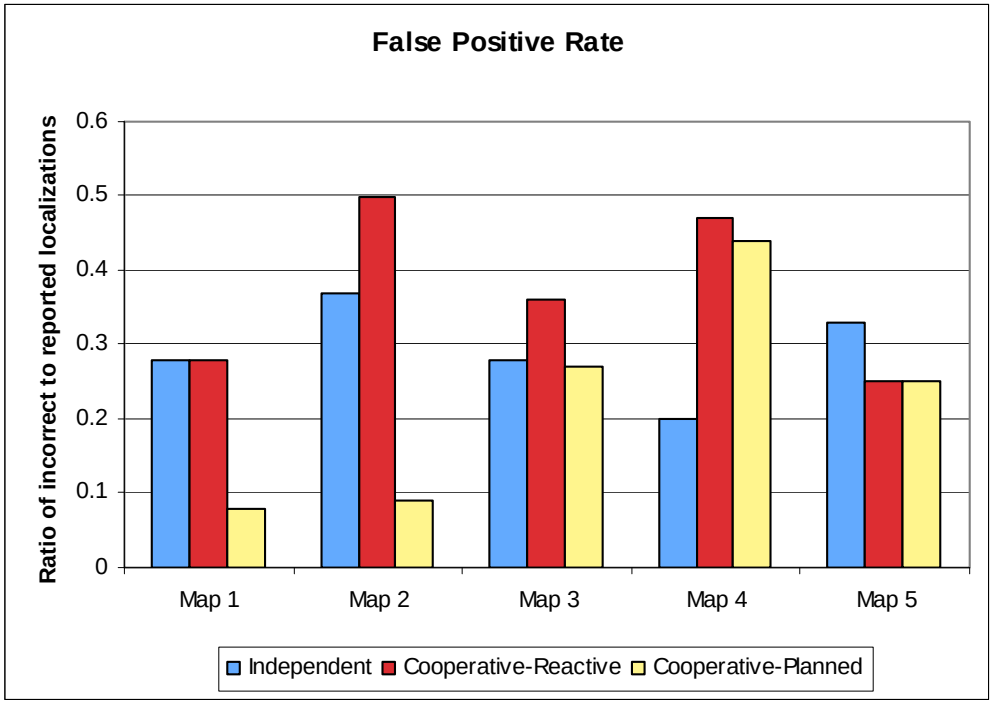

Figure 34: False Positive Rate

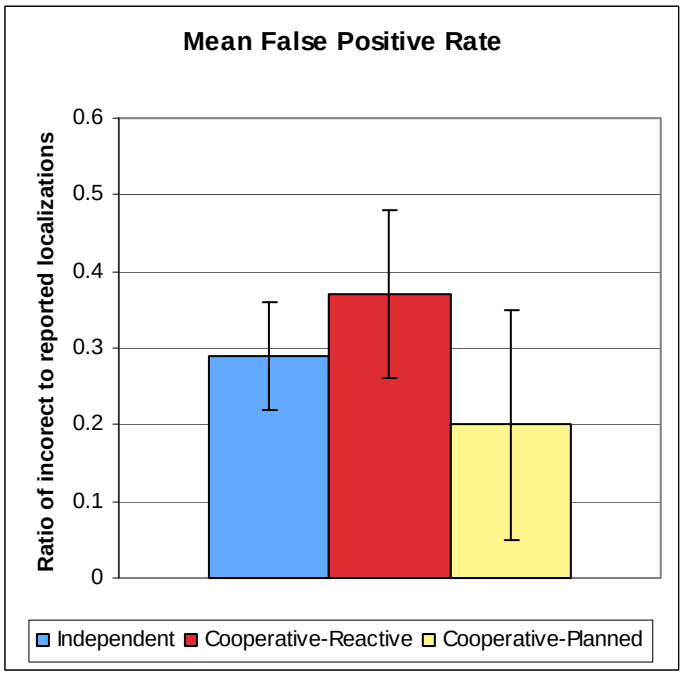

Figure 35: Mean False Positive Rate, with standard deviations shown.

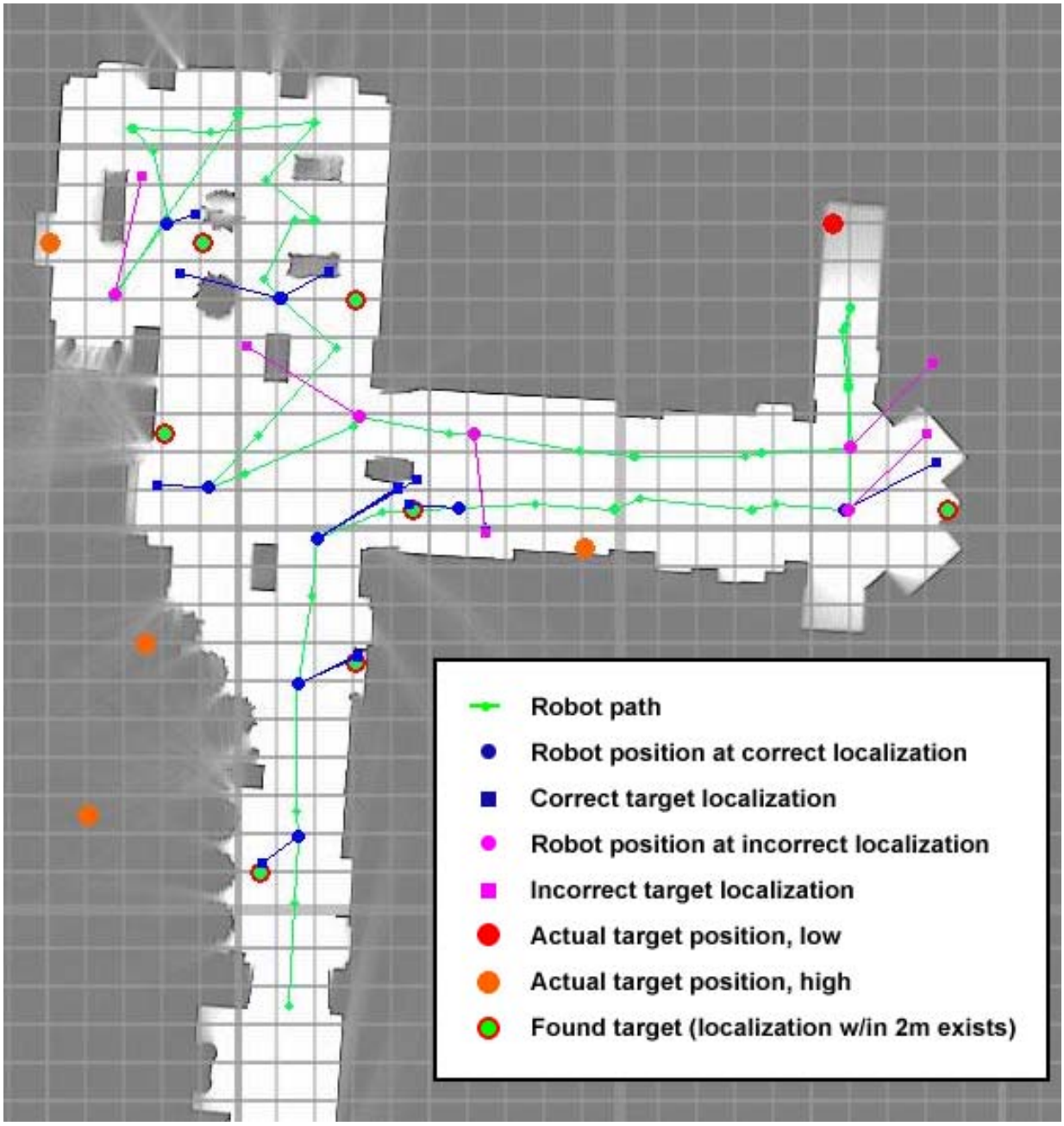

Figure 36: The path of the robot showing target localizations along path for Map 3, Independent. The following list explains the symbols used and what they represent:
Green points + green lines represent the path of robot.
Red circle represents an actual target position, low.
Orange circle represents an actual target position, high.
Orange or red circle with green dot in center represents a detected target.
Blue circle represents the position of robot at a successful target localization.
Purple circle represents the position of robot at an incorrect target localization.
Purple square represents the reported position of a target at incorrect localization.

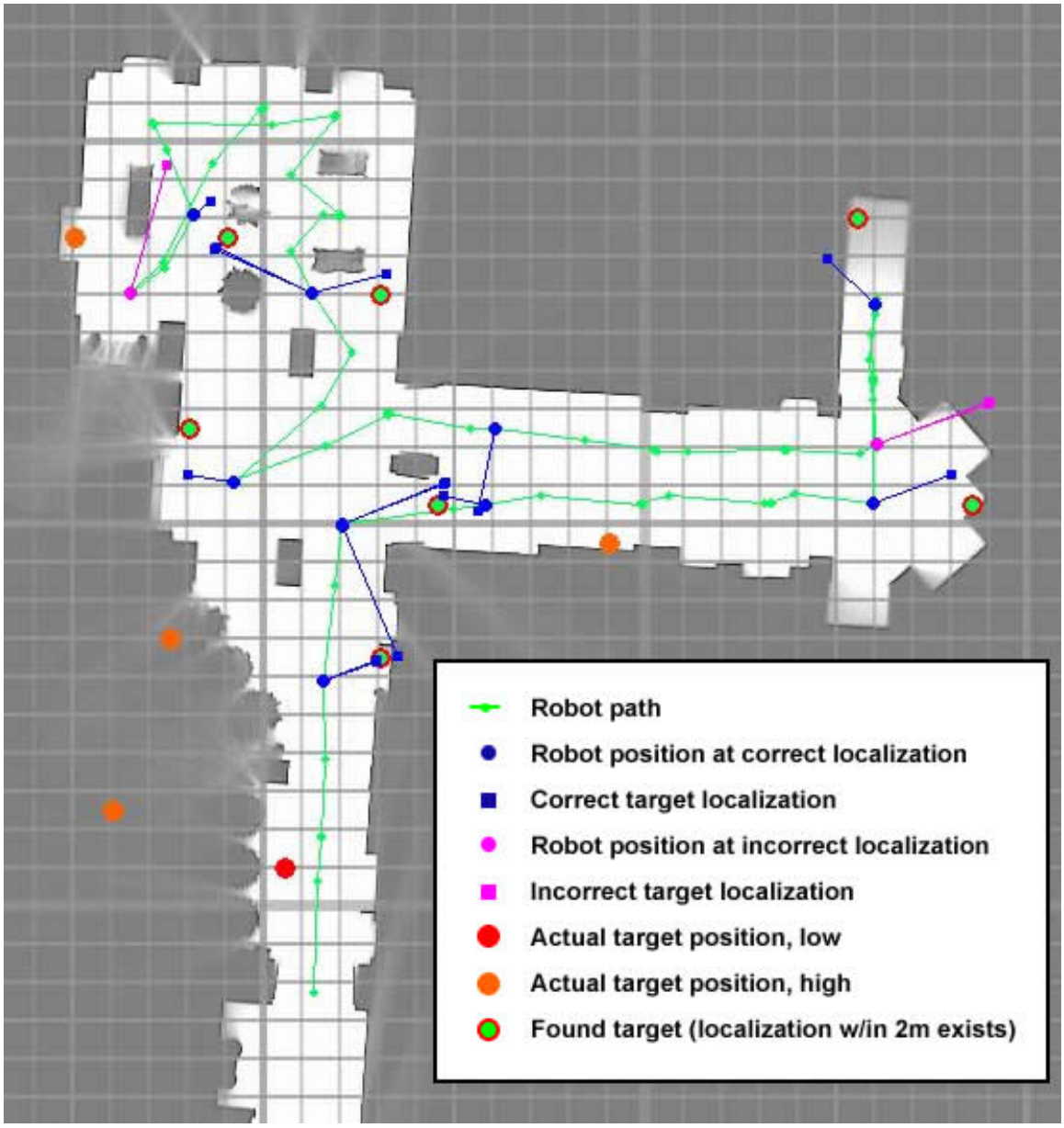

Figure 37: The path of the robot showing target localizations along path for Map 3, Cooperative Planned.
The following list explains the symbols used and what they represent:
Green points + green lines represent the path of robot.
Red circle represents an actual target position, low.
Orange circle represents an actual target position, high.
Orange or red circle with green dot in center represents a detected target.
Blue circle represents the position of robot at a successful target localization.
Purple circle represents the position of robot at an incorrect target localization.
Purple square represents the reported position of a target at incorrect localization.

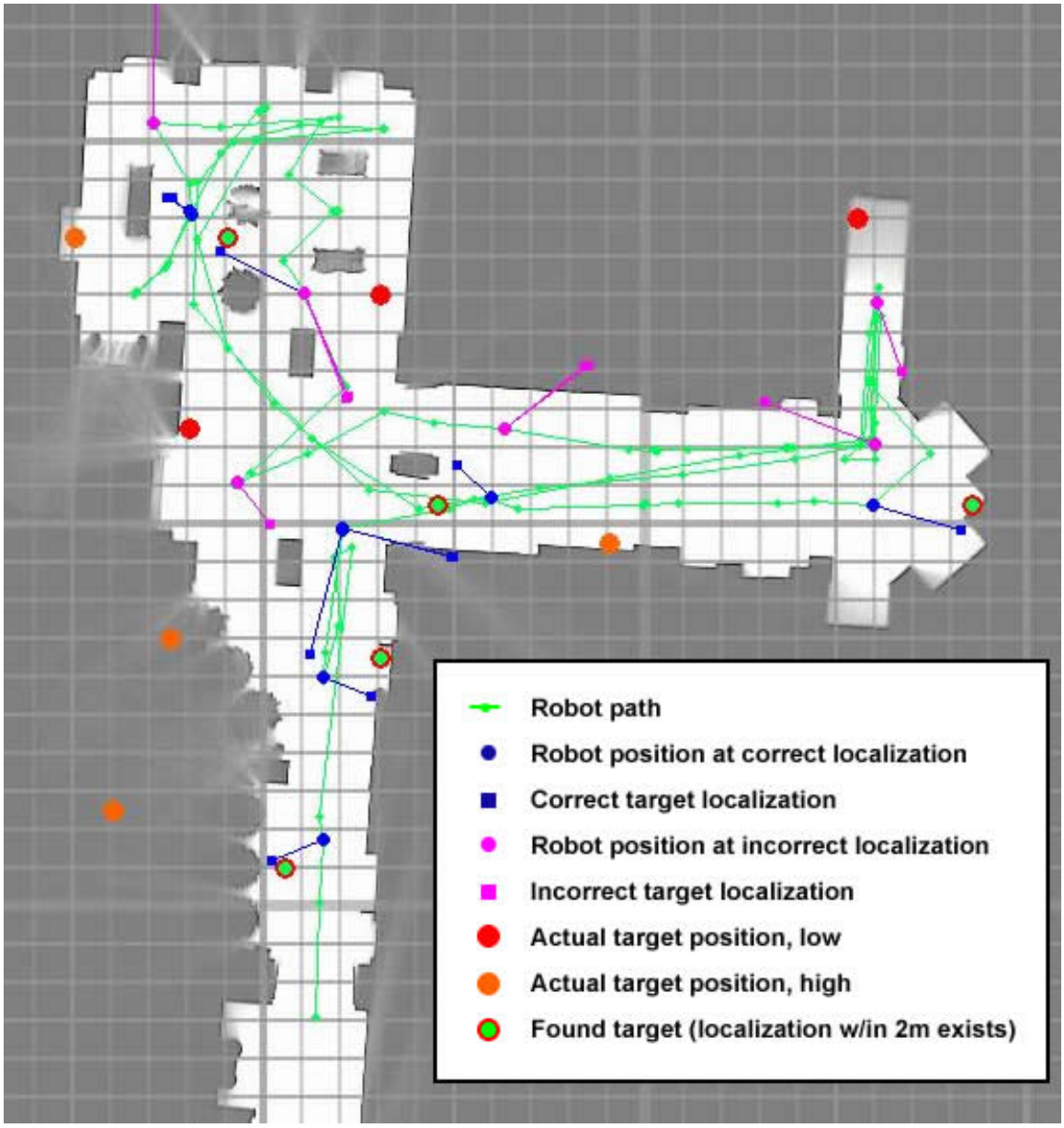

Figure 38: The path of the robot showing target localizations along path for Map 3, Cooperative Reactive.
The following list explains the symbols used and what they represent:
Green points + green lines represent the path of robot.
Red circle represents an actual target position, low.
Orange circle represents an actual target position, high.
Orange or red circle with green dot in center represents a detected target.
Blue circle represents the position of robot at a successful target localization.
Purple circle represents the position of robot at an incorrect target localization.
Purple square represents the reported position of a target at incorrect localization.

error is not accumulating. I do not believe this increase in the rate of false positives is in any way insurmountable; however, it is worth noting and future work should address this issue where necessary.

5.4 Summary

To compare the approaches in terms of these findings, the Cooperative Reactive and Planned approaches both perform the task of target localization equally well in terms of Target Localization Accuracy, and better than the Independent approach. In essence, if there is any benefit at all to be had from cooperation, the cooperative behaviors will realize this benefit whereas the Independent will not. In terms of Task Completion Time, the Cooperative Planned performs slightly better than the Cooperative Reactive, with the caveat that the magnitude of the performance increase is directly related to the speed of the robot. However, if one takes into account the energy expended by the robots by considering the Aggregate Run Time, the Cooperative Reactive is superior to the Cooperative Planned.

When choosing an approach, it may also be necessary to consider the values placed on human time and robot time. For example, in some situations human time may be highly valued, so an approach that minimizes the human-controlled UAV run time may be favored.

Taking all of those considerations in mind, I would suggest this solution: In any scenario that involves a task that would benefit from cooperation part of the time and human and robot time are valued equally, if there is a demand for the most rapid completion of the task possible, a Cooperative Planned approach should be employed. On the other hand, if energy consumption or aggregate run time of the robots is a concern, or in a situation where a planned approach is not possible, a Cooperative Reactive approach might be best. In a scenario where human time is valued much higher than robot time, even though a Cooperative Planned approach is best in terms of overall Task Completion Time, a Cooperative Reactive approach might be preferred because it reduces Aggregate Run Time by reducing the human-controlled UAV run time.

I also observe that for some mission configurations, especially those with a high number of critical junctures, the increased cost of a reactive cooperation should be taken into account, especially the higher potential for false positives and the typical distance between robots that must be constantly traversed, and a Planned Cooperative approach should be considered instead.

6. Conclusion

In this work I have leveraged the power of ASyMTRe to identify critical points at which tightly-coupled cooperation/coordination benefits a heterogeneous team of less-than-fully-capable robots. Where previous related work [8] had identified these critical points manually, I have identified them algorithmically, both a priori and reactively during task execution. Where the previous related work's approach to synergistic cooperative behaviors had been validated in simulation, I have validated the benefit of cooperation at these critical junctures in real-world experimentation.

While both a priori Planned cooperation and Reactive cooperation have been shown to perform identically well in terms of the target localization task, I have examined differences that distinguish Planned vs. Reactive, including Task Completion Time, where Planned slightly outperforms Reactive, Aggregate System Run Time, by which Reactive performs better than Planned, and False Positive Rate, by which it appears Planned performed better than Reactive.

Because of these findings, it appears that a Cooperative Reactive-type approach would be best for situations where knowledge for a planned approach is not possible or energy use or aggregate run time (e.g., wear and tear on the robots) is a significant factor, unless a large number of critical junctures are present or expected, which could increase the rate of false positives. Also, the speed of the robot should be taken into account when deciding whether to use a Cooperative-Reactive approach over a Cooperative Planned; a faster robot may make the Cooperative Reactive approach more cost-effective in terms of Task Completion Time. A Cooperative Planned-type approach would be best for situations where a large number of CJs are expected and a higher false positive rate is unacceptable, or where a small increase in time performance is preferred.

References

[1] http://j.piri.home.mchsi.com/parts.htm

[2] http://playerstage.sourceforge.net/

[3] http://rcguys.com/

[4] http://www.cheapbatterypacks.com/

[5] http://www.rctoys.com/

[6] http://www-robotics.usc.edu/~ahoward/pmap/

[7] R.C. Arkin. Motor schema based navigation for a mobile robot: an approach to programming by behavior. *Proceedings of the IEEE Conference on Robotics and Automation*, 264–271, 1987.

[8] H. Choxi, C. Bolden Distributed Control for Unmanned Vehicles. Lockheed Martin Advanced Technology Laboratories unpublished internal paper, 2005.

[9] B. R. Donald. Information invariants in robotics. *Artificial Intelligence,* 72:217–304, 1995.

[10] T. Fong, I. Nourbakhsh, C. Kunz, L. Fluckiger, J. Schreiner. The Peer-to-Peer Human-Robot Interaction Project. *Proceedings of AIAA Space*, 2005.

[11] B. P. Gerkey, R. T. Vaughan and A. Howard. The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems. *Proceedings of the International Conference on Advanced Robotics*, 2003.

[12] A. Howard, L. E. Parker and G. S. Sukhatme. The SDR Experience: Experiments with a Large-Scale Heterogeneous Mobile Robot Team. *Proceedings of the 9th International Symposium on Experimental Robotics,* 2004.

[13] E. G. Jones, B. Browning, M. B. Dias, B. Argall, M. Veloso, A. Stentz. Dynamically formed heterogeneous robot teams performing tightly-coupled tasks. *Proceedings of the IEEE International Conference on Robotics and Automation*, 570 – 575, 2006.

[14] D. Jung, A. Zelinsky. Grounded Symbolic Communication between Heterogeneous Cooperating Robots. *Autonomous Robots*, 8(3) 269 – 292, 2000.

[15] D.M. Lyons, M.A. Arbib. A formal model of computation for sensory-based robotics. *IEEE Transactions on Robotics and Automation*, 5(3):280–293, 1989.

[16] J. Marble, D. Bruemmer, D. Few, D. Dudenhoeffer. Evaluation of Supervisory vs. Peer-Peer Interaction with Human-Robot Teams. *Proceedings of the Hawaii International Conference on System Sciences*, 2004.

[17] L. E. Parker. The Effect of Heterogeneity in Teams of 100+ Mobile Robots. *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata*, ed. by A. Schultz, L. E. Parker, F. Schneider, 2003.

[18] L. E. Parker. Multiple Mobile Robot Systems. Chapter 40 of *Springer Handbook of Robotics*, by B. Siciliano, O. Khatib (Eds), Springer-Verlag, 2008.

[19] L. E. Parker, B. Kannan, F. Tang, M. Bailey. Tightly-Coupled Navigation Assistance in Heterogeneous Multi-Robot Teams. *Proceedings of IEEE International Conference on Intelligent Robots and Systems,* 2004.

[20] L. E. Parker, F. Tang. Building Multirobot Coalitions Through Automated Task Solution Synthesis. *Proceedings of the IEEE*, 2006.

[21] F. Tang, L. E. Parker. ASyMTRe: Automated Synthesis of Multi-Robot Task Solutions through Software Reconfiguration. *Proceedings of IEEE International Conference on Robotics and Automation,* 2005.

[22] F. Tang, L. E. Parker. Peer-to-Peer Human-Robot Teaming through Reconfigurable Schemas. AAAI Spring Symposium on "To Boldly Go Where No Human-Robot Team Has Gone Before," March 2006.

Appendix

Appendix A-1: ASyMTRe info.cfg file. This file defines the perceptual schemas (ps), communications schemas (cs), and information types (f) available, and their interactions. This information is used by the ASyMTRe reasoning system to map environmental sensors with perceptual, motor, and communication schemas.

```
# ps1: calculates self global position
# ps2: calculates self goal position (hard-coded)
# ps3: calculates global position of another agent
# ps4: calculates self global position according to detected relative
#      position of another agent and global position of the other agent
# ps5: calculates the relative position of another agent
# ps6: calculates relative position of EDI
# ps7: calculates global position of EDI using relative position, self
#      global position
# ps8: calculates self global position using relative position of EDI,
#      global position of EDI
# ps9: "looks up" global position of EDI

# cs: transfer information between schemas

# f1: self global position
# f2: other global position
# f3: other's relative position
# f4: EDI global position
# f5: EDI relative position

goal {goto}

provide {
  ps1 > f1
  ps3 > f2
  ps4 > f1
  ps5 > f3
  ps6 > f5
  ps7 > f4
  ps9 > f4
  ps8 > f1
}

need {
  goto < f1
  ps4 < f2 & f3
  ps3 < f1 & f3
  ps8 < f4 & f5
  ps7 < f5 & f1
}

communicate {
  f1 < cs > f2
  f2 < cs > f1
}
```

Appendix A-2: ASyMTRe commonsense.cfg file. This file contains the costs associated with the use of each schema. Using this information combined with the flow of information defined in info.cfg (Appendix A-1), a task solution is generated. The composite is the last configuration solution generated and is saved at the end of this file.

```
# ps1: calculates self global position
# ps2: calculates self goal position (hard-coded)
# ps3: calculates global position of another agent
# ps4: calculates self global position according to detected relative
#      position of another agent and global position of the other agent
# ps5: calculates the relative position of another agent
# ps6: calculates relative position of EDI
# ps7: calculates global position of EDI using relative position, self
$      global position
# ps8: calculates self global position using relative position of EDI,
#      global position of EDI
# ps9: "looks up" global position of EDI

# cs: transfer information between schemas

# f1: self global position
# f2: other global position
# f3: other's relative position
# f4: EDI global position
# f5: EDI relative position

atomic {
  ps1 & gps = 95
  ps1 & laser = 95
  ps3 & dummy = 100
  ps4 & dummy = 100
  ps5 & laser = 95
  ps5 & camera = 90
  cs & comm = 100
  ps6 & camera = 90
  ps6 & laser = 95
  ps7 & dummy = 100
  ps8 & dummy = 100
  ps9 & dummy = 100
}

# Cost description
cost {
  gps = 1,
  comm = 20,
  camera = 2,
  laser = 3,
  dummy = 1,
}

composite {
1,
goto = 8,
cs & [cs & f1] |
```

```
ps1 |
cs & [cs & f2] & ps4 & ps5 |
cs & [cs & f1] & ps3 & ps4 & ps5 |
ps1 & ps3 & ps4 & ps5 |
cs & [cs & f1] & ps6 & ps7 & ps8 |
ps1 & ps6 & ps7 & ps8 |
ps6 & ps8 & ps9
}
```

Appendix A-3: Pseudocode for robot Cooperative Reactive and Independent behavior. Cooperative Planned is the same, except at certain waypoints, the (UAV_message is "help") condition returns true by default.

```
Given a set of waypoints

for (each waypoint) {
      moveto_waypoint()
      while not at waypoint {
            helped = check_if_help_needed()
            if (helped) load last waypoint
      }
      if (!helped) target_scan()
}

function check_if_help_needed() {
  UAV_needs_help = false
  robot_helped_UAV = false
  do  {
      UAV_message = check_message()
      if (UAV_message is "help") {
            UAV_needs_help = true
            moveto_waypoint()
            robot_helped_UAV = true
            }
      else if (UAV_message is "resume")
            UAV_needs_help = false;
      } while UAV_needs_help
  return robot_helped_UAV
}

function target_scan(){
number_of_scans = 10
for (number_of_scans) {
      turn 360 / number_of_scans degrees clockwise
      target_detect()
      }
return to original orientation
}

function target_detect() {
for (each blob detected) {
      check if target area is within threshold
            if not within threshold, next blob
      check if target relative dimensions are within tolerances
            if not within tolerances, next blob
      center target in frame (to get it fully in-frame)
      calculate target position using area of blob and orientation of
robot
      return to previous orientation
      }
}
```

Vita

Christopher Michael Reardon earned a B.S. in computer science with a minor in biology from Berry College in 2002. He is currently pursuing his Master of Science degree in computer science at the University of Tennessee, Knoxville.