1. Background

Like in the animal kingdom, the world of unmanned systems has vehicles for land, sea, and air. The unmanned aerial vehicles (UAVs) have the largest piece of the pie when it comes to research dollars. The unmanned ground vehicles (UGVs) take the next largest piece, which leaves the vehicles of the sea, including unmanned surface vehicles (USVs) and unmanned underwater vehicles (UUVs), splitting the smallest piece of the research funds. As a result of this division of research, UAVs and UGVs are leading the way in terms of autonomous capabilities. This project intends to provide some progress in the realm of UUV autonomy with a particular task at its core.

The Center for Marine Biodiversity & Conservation (CMBC) is a part of the Scripps Institution of Oceanography. This organization does underwater research all over the world. Often the research involves the emplacement of underwater sensors, which will collect data for years at a time. There are many problems with such long emplacement times. The most obvious problem is the lags in analysis and further progress while waiting on sensor data. Also, if a sensor breaks while emplaced, the researchers will not discover the problem until they finally retrieve the sensor and attempt to download the data.

There is great potential here for a semi-autonomous UUV to navigate to the sensors and download the data. Depending on the breadth of the sensor field, the UUV could do these sweeps monthly, weekly, or potentially even daily. This will provide the researchers a steadier stream of data to work with and therefore allow for much more flexibility to alter aspects of the experiments.

2. Project Goal

The vehicle that I am using for this project is called the Stingray. This vehicle is owned and maintained by a non-profit organization called San Diego iBotics Student Engineering Society (SD iBotics). The vehicle was originally designed for use in the Association for Unmanned Vehicle Systems International (AUVSI) International Autonomous Underwater Vehicle Competition and is well suited for such a project. Also, I am currently working with the iBotics team in preparation for the 2009 AUVSI Competition, so there may be some overlap of resources and progress.
The goal for this project within the scope of the class is to work with the navigation techniques available on the Stingray and attempt to get local position. There are many different techniques for underwater navigation. Some are more effective in certain scenarios, such as deep water, while others can be used in multiple environments. Some of these techniques are improved by working in collaboration with other techniques. And some can only be used on vehicles of a certain size or with a certain capability. An overview of these techniques can be seen in the Literary Survey document for this project.

The navigation techniques that I have access to for this project are limited however. First I will be able to do a minimal dead reckoning system using an inertial navigation system (INS) and a pressure sensor for depth. The other technique is computer vision based on an a priori map of the environment. I will do this local positioning within the context of the AUVSI Competition described above.

3. Approach

The preliminary steps to beginning this work include gaining subversion server accessing, setting up the correct operating system, installing team preferred development environment, and getting the code to compile. Also, I need to understand the flow of data through the code and how different modules interact. Finally, I need to prepare a module or section of code where the positioning effort makes sense to reside.

Once everything is prepared, the real work can begin. There is currently no mechanism for calibrating the SSI Technologies pressure sensor in order to receive accurate depth data. So first I need to prepare this, as it will be useful in both a dead reckoning system, and in a computer vision system. After the pressure data is converted into a depth, I can prepare a Kalman filter to merge the INS data, depth, and motor commands to gain a running estimation of position based on starting point.

The problem with dead reckoning is that the errors compound with time. So the longer the system runs, the worse the position estimation will be from my Kalman filter. This is where the computer vision element becomes important. The position system will use the estimation from dead reckoning and the current direction of travel to decide what vision search to do. When the vision finds known objects from the a priori map, then it will adjust the dead reckoning position. The computer vision technique will, in effect, be a reset mechanism for the dead reckoning, causing the error to restart as close to zero as possible.

The final piece of my approach is to set up a visualization tool that can be used to evaluate the results of the positioning system. This visualization will be a map
that is to scale of the environment for the Stingray. Then as the Stingray is manually maneuvered through the environment, and icon will be displayed on the map. This will allow us to compare real location to visualized location to determine accuracy.

4. Progress

I have spent a lot of time install Virtual Box onto my MacBook Pro, and installing Ubuntu 8.10 Intrepid onto the virtual machine. This was quite a process to get everything running correctly, and linking the networking through OS X. In the end I needed three different network adapters for wireless, Ethernet, and a special case for UCSD wireless, because of the way DHCP is done. I have finally gained access to the iBotics subversion server and have checked out the code. I also have installed the development environment called Geany.

The initial setup of my laptop for work on this project has been a substantial effort so far. In addition, I have spent time with iBotics developers learning about the modules of the Stingray software and working with the team on testing. Knowing how the modules interact is essential for gaining access to the sensors needed for my local positioning.

In order to understand the work that I have done and the future work, I will describe the modules. The navigation module controls the five propellers on the vehicle: one in the tail for pitch, two in the wings for roll, and the two voith propellers underneath for yaw. This module is constantly aiming for a target, which is defaulted to a flat and steady position. This means that if you push down on one of the wings or the tail, the Stingray will right itself. Changing this target is what causes the vehicle to move, which can be done from other modules.

The vision module uses the two cameras, one in front and one underneath, to capture and process frames based on the current task. The task determines what algorithm the vision module uses on the image frames, and in effect determines what the vision module is looking for. This task element can be changed from other modules.

There are two such modules that generally will change the target in the navigation module and the task in the vision module. The one that is actually running on the Stingray is called the planner module. This is the module that controls the autonomy of the vehicle. It tells the vision module what to look for through tasks and then creates targets for the navigation module based on the vision data and other sensor data.

The second module that can control the navigation targets and the vision tasks is the Dock Control Station (DCS), which is the GUI. This module provides statistics on the vehicle INS, pressure sensor, etc. It also allows a user to manually control the
vehicle as the planner would. It can tell the planner module what task to run, which in turn controls the vision module, and it can tell the navigation module where to go.

There are also, some other modules that are not considered main modules. One of these is the joystick module, which allows for control of the vehicle via gaming controller. Another important module is the labjack daemon, which monitors the battery levels on the vehicle and also captures the pressure sensor data. Currently the labjack daemon is connected to navigation, and the data is packaged in the navigation status.

These modules are separate processes that communicate via TCP connections. This means that I could potential create a new module called local position system (LPS) module to estimate position. This module would be able to connect to all the modules it needs data from, and then simply process the data through the Kalman filter and vision algorithm and provide a location out to other modules. The module that would want this information in the long run is planner, because this would be useful in deciding what task to do next. For the scope of this project, the LPS modules position estimation will be used by DCS to visualize the location.

So far my work in the Stingray code has been to change the way tasks and targets were created. Prior to my work, the navigation module received the vision data and created a target to aim for. Also, the vision module was continuously looking for multiple objects, or the code was commented so that there was only one task executing at a time. As described above, I changed this so that the planner module is now in the loop controlling the tasks and creating the targets based on the vision data. The navigation module no longer even connects to the vision module, which saves on network traffic. I also connected this to the DCS, so now during tests we can easily change the vision task and see if the vehicle reacts appropriately. This will help with debugging the vision algorithms and the targeting.

This work described is not directly useful for my project, but it was a chance to learn the code and bring the planner into the system. It was necessary to work with the modules in a way so that I could see if my results worked. Now I am confident with each module and can move forward with my efforts.

5. Future Work

I am currently working on a mechanism to calibrate depth. Right now, the pressure sensor provides a PSI through the labjack board, but this needs to be converted into units of distance. To do this we basically need to be able to manually capture and log PSI readings. The initial process will be to manually operate the Stingray to a depth measured by a yardstick or a rope, and the capture the pressure sensor data. Analyzing this data will help us determine the best way to have a more automated
process for calibrating the depth sensor. I intend to do this data capturing during water testing of the controllable tasks on Friday May 8th.

Next I will write my LPS module and connect it to the navigation and vision modules. This task will be complete when I can run the navigation, vision, LPS and DCS modules, and have LPS receive sensor data from navigation and vision data, and provide these to the DCS for display.

After I have a working module for doing positioning, I will implement a Kalman filter to get a single status from the INS, depth, and motor controller data. This task will be complete when I can run all the modules and see a single value from the LPS module displayed in the DCS. It will then take some testing to verify the results of this filter. Testing will involve refreshing the positioning and then manually driving the vehicle and logging the position estimation from the Kalman filter. Then I will compare the changing relative positions to what I actually saw in the water. Likely this evaluation process will be open for improvements along the way.

Finally, I will attempt to create vision techniques to try and localize the position of the vehicle based on objects in the screen. This part of the project needs some more research before implementation, and may not be realized during this quarter. Once am to the point of beginning this, I will reevaluate the time remaining and potentially adjust my goals.

One additional goal that may replace the vision, or may be added after depending, is a visualization technique. This would be an image of the pool and the obstacles to scale, so that the estimated position can be visualized on the map. This will allow for a much better evaluation of the quality of the LPS module.

6. Conclusion

I believe strongly that the results of this project will be useful for the iBotics team and provide improved autonomy in the planner module. Even beyond the AUVSI competition, which is the main purpose of the iBotics effort, there is great potential for real world applications such as in Moorea as described in the background. Therefore, the work on this project will continue even after this quarter ends, which means that whatever I accomplish in this class will be a head start for the future efforts.