Design Report

Robot's Name: CATA (City Autonomous Transportation Agent)

Team Members:

- Carlos Jaramillo
- Sebastian Pendola
- Achyut Shrestha
- Carlos Chinchilla

Required Faculty Advisor Statement:
I certify that the engineering design and improvements since last year on the ground vehicle described in this report by the CATA team have been significant and for which each team member has worked on the project the equivalent amount of four semester hours of a project/design course during the Spring 2011 term.

CCNY ROBOTICS LAB
Est. 2002

Xiao Ji

Professor Jizhong Xiao
Department of Electrical Engineering, City College

Table of Contents

I. INTRODUCTION

The City College of New York participated last year in the 18th Annual Intelligent Ground Vehicle Competition (IGVC 2010) using the City-ALIEN robot, which succeeded mainly as a proof of concept for an outrageous design based uniquely on mirrors and computer vision. This type of application is known as "***cata****dioptrics*," a composed word from the Greek "*catoptrics*" meaning "shaped mirrors" and "*dioptrics*" meaning "lenses" (in this case from a camera). This time, we are dubbing our robot ***CATA***, which is also an acronym for ***C****ity* ***A****utonomous* ***T****ransportation* ***A****gent.* Thus, the robot's name implies its functionality as an autonomous mobile vehicle that can transport a "load" to a particular point in the map (usually given by GPS coordinates outdoors or by knowing the position on an existing/evolving map indoors).

Although the chassis of the robot has remained unchanged (except for the new license plates), all mechanical parts such as wheels and motors, and electronic components have been upgraded. We have decided to make navigation more robust by adding a low-price laser range-finder (a Hokuyo URG), without affecting the aesthetics of the old design based purely on vision. Indeed, a great improvement from last year is the totally redesigned electrical system adopting a "common ground" with a simplified architecture that increases safety in various ways. However, the most important change this year is the entire software architecture based on the emerging and open-source *Robotics Operating System* (ROS) running under GNU/Linux, as opposed to the old LabVIEW system in Windows used in past years. With these aspects in mind, our design summarizes to the following:

> ***Design Goal:*** *Simplification, safety, and portable modularity of an autonomous ground vehicle with pure open-source software.*

The remainder of this document provides an extent information about CATA's hardware and software architectures. A quick view of the innovations this year is given next.

Summary of Improvements

Electrical Unit

- Safer charging method to protect components while charging and operating.
- Very bright LED indicators controlled via Arduino board wtih ATMEGA 328 microcontroller interfacing with ROS. Indication of operation mode via lights.
- Simplified power distribution and electronics connectivity: Less is more!
- Using a MOSFET and relay in Emergency Stop (e-Stop) wireless mechanism.

Modular Software Architecture with ROS

- Portability and re-usability of software development
- Message passing standardization
- Benefits from an open-source community
  i. Saving costs of software licensing (LabVIEW was ugly expensive)
  ii. Dynamic programming with centralized and distributed repositories
  iii. Excellent visualization tools (rviz) and synthetic robot modeling (URDF)
  iv. Offline data collection (rosbags): allow us to work offline with real sensor data by recording it and replaying it
- Voice feedback to communicate operational mode and dynamic changes at run-time from joystick and embedded autonomous navigation processes.

Map-based Navigation:

- Constructing 2-D local and global costmaps makes path-planning easier.
- Lane detection method is no longer trying to fit lines, but we instead segment (threshold) color of lanes and mark them as obstacles in the map.

II. HARDWARE

Electrical System

The electrical system, designed from scratch two years ago, went through a major redesign to increase the safety of the vehicle and its devices. The following changes were made.

- Addition of common ground and grounded frame.
- Replacement of metal enclosures by isolated plastic enclosures.
- Division of voltage with batteries.
- Replacement of light bulb by LEDs for safety light.
- Electrical isolation of wireless E-Stop.
- Replacement of relays and PIC by AX2850 Motor Controller.
- Using 348-Series Bison DC brushed motors.

The following sections contain detail descriptions of the electrical subsystems as well as the changes from previous years of the subsystem described.

Common Ground & Grounded Frame

The vehicle's metal frame is connected to the common ground, which is connected to the negative side of one of the two 12V batteries connected in series. This type of connection is safer than in previous years designs because it reduces the risks of short-circuiting.

Isolated Electrical Enclosures

Two plastic enclosure boxes replaced the metal terminal enclosure used in previous years. The smaller plastic box carries the common ground terminal and is sealed for safety purposes. The larger plastic box carries the 12V terminal on one side, the 24V terminal on the opposite side, and other connections in the middle. The use of plastic over metal reduces the risks of short-circuiting. Besides, adding proper insulation to exposed wires, the external plastic semi-sphere, also known as the mirror dome, houses all the electrical components. It protects the electrical system from hazardous weather conditions such as rain and extreme sunlight (heat).

Division of Voltages

The two 12V batteries are connected in series to provide a 24V terminal while the 12V terminal is created from a single battery (the battery adjacent to ground). In previous years an ATX took care of this voltage division but violated the common-ground design.

Power Distribution

Two 40-Amp-Hours 12VDC sealed lead acid batteries are connected in series supplying 24VDC to the ATX, the power inverter, and the motors through the motor controller. The ATX provides power to the wireless router, laser, and the safety light. The power inverter powers the GPS system, the on-board computer, the compass, and the cameras. The battery adjacent to ground provides 12V to the wireless E-Stop receiver, and the motor controller.

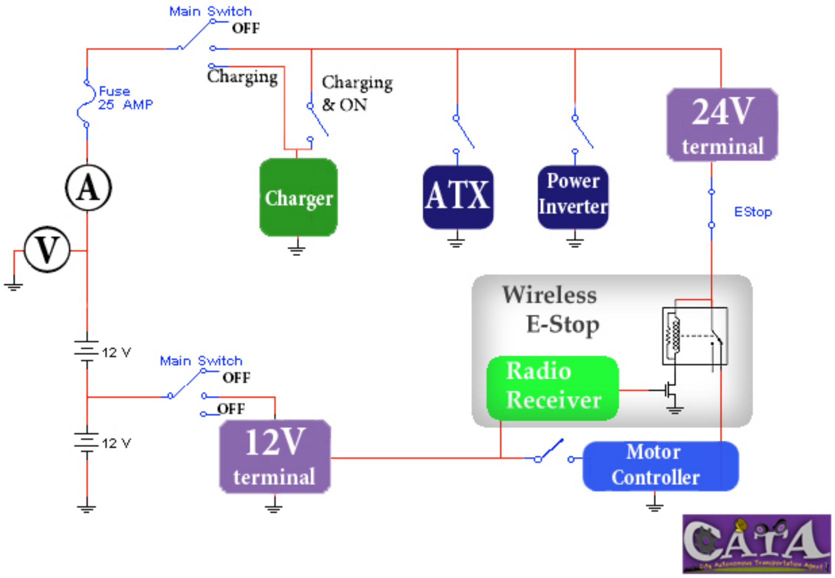


Figure 1. Schematic of CATA's electrical system

Power Consumption

For convenience, the 500W battery charger has a switch that allows powering the electronic devices while charging. However some features are disabled, like the mobility of the vehicle, and wireless E-Stop.

The following data pertaining battery life is taken from usage and practice runs, as well as other experimentation.

Mode	Time (system ON)	Time (system OFF)
Charging	~7 hours	~5 hours
Driving	~6 hours	NA
Stationary	~8 hours	NA

Power Switches and Indicators

An ammeter and a voltmeter are placed on the left side of the robot, next to the main power switch and plug for the battery charger. The main switch has three states:

On, the robot is fully functional;
Off, all power is disconnected; and
Charging mode.

There are two options enabled by a switch when in *charging* mode; charging only, and charging and using only the devices attached to 24VDC.

Power Supply to AC components

A PowerBright ML400-24 24V DC to 120V AC Power Inverter (400 Watt Capacity) provides AC power to the laptop charger, and other off-the-shelf components that can be operated via their AC adapters without the need of customizing their power cables.

Emergency Stop System

The wireless and button E-Stop are connected in series on the 24V terminal. When either one is pressed, the power to the motors is cut off thus bringing the vehicle to a quick full stop. Cutting the power only to the motors avoids damaging other electronic devices connected to the electric system.

Button E-Stop

A normally-open red button on the rear of the vehicle cuts off the power to the motors when pressed. In such state, other electronic devices, not directly related to the motion of the vehicle, remain turned on to avoid electrical damage or loss of configuration.

Wireless E-Stop

Consists of a radio transmitter and receiver, a MOSFET and a relay. The radio transmitter and receiver work in the 310MHz AM frequency with a range over 250 feet. To avoid unwanted feedback generated by the inertia of the motors rotation, the MOSFET and radio receiver are connected to a different voltage than that of the motors. The radio receiver controls the MOSFET, which in turn drives the relay that switches the power to the motors on or off.

Safety Light

A series of LEDs in series replaced the flashing light bulb used in previous years. The safety light is formed by a set of ten LEDs in series driven by an Arduino board with an ATMEGA328 microcontroller. The microcontroller communicates with the on-board computer, which in turn commands the way in which the LEDs should flash given the mode of operation of the robot. The microcontroller is programmed to flash the LEDs in different patterns.

Motor Control

Motors

We have replaced the old motors with new 348-Series BISON 24VDC brushed motors. Each of the two motors has an impedance of 0.8 Ohms and can take a maximum current of 30A.

Wheel Optical Encoders

The US Digital Incremental Rotatory E5 (Part No. E5-32-188-I-S-H-T-B) are quadrature encoders that read the number of pulses per revolution (set to 1000 PPR). Odometry (wheels' physical displacement) is predicted from the left and right wheels' angular velocities interpreted from their encoder readings. The motor controller powers the encoders with 5V.

Motor Controller

The AX2850 Motor Controller by ROBOTEQ replaced the PIC and relays formerly in charge of controlling the motors direction and power distribution. The current motor controller is capable of performing odometry through the two encoders attached to each motor. In addition, the on-board computer commands and communicates with the motor controller through ROS. 12V taken from the 12V-terminal power the motor controller's microcontroller on. For safety, this power is not interrupted when either of the E-Stop systems is triggered. The two motors are powered with 24V that are input to the motor controller.

III. SOFTWARE

Introducing the ROS Framework for Autonomous Navigation

The Robotics Operating System (ROS) is an open-source robotics framework with an active community and a vast number of software packages. The City College of New York (CCNY) Robotics lab is an active contributor to the ROS project. We believe that among the strongest reasons that justify adopting ROS are among the following:

> **Modularity:** “ROS encourages a "**divide and conquer**" strategy by wrapping each task into a separate node rather than building one large program that can be hard to understand or debug.”
>
> **Portability & Re-usability**: We do not want to have to re-invent the wheel every time we program a new robot. Thus, given that the hardware layer is properly abstracted via ROS drivers, the same algorithms and functionality can be provided to any robot that is able to run ROS.

The above paragraph briefly introduces the concept on “nodes”, which is just a way to call an executable module in the ROS world. Since ROS is an open-source project, we have taken advantage of the large collection of documented APIs (application programmable interface) and libraries to construct and provide the desire autonomous behavior that CATA requires for the IGVC autonomous navigation challenges.

Software Architecture

ROS has a well-defined collection of “Message Types” that standardize the communication among several “nodes” or processes. In our particular case, we require that **sensors**, such as encoders (odometry), GPS, Inertial Motion Unit (compass), laser, and cameras, can all communicate with the ROS server (running on robot) through drivers. This information is thus formatted to specific message types that other nodes (processes) of interested, such as map building and obstacle avoidance algorithms, can subscribe to and utilize the data. Eventually, the same processes are fused (filtered) to provide localization and plan a trajectory to goals, and eventually, the desire commands, such as velocity, are sent to the actuators (motors).

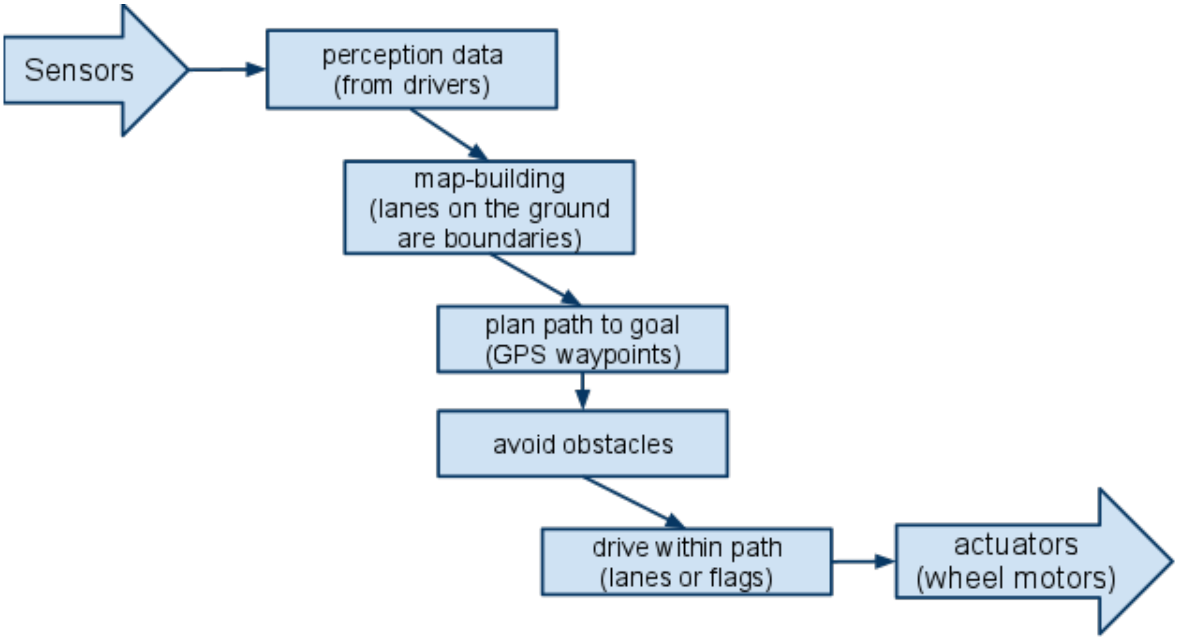


Figure 2. Architecture for map-based navigation

Map Building

Using *gmapping* with ROS, we are able to produce grid maps from laser range data. Unfortunately, the maps produced are only 2-D and need as input laser-range data, which come from the actual laser rangefinder and the stereo camera point-cloud sliced and converted into laser message type. At the moment we are able to make indoor maps that fulfill loop-closure. Figure 3 is an example of such maps using *gmapping*.

Figure 3. Example of an indoor map using *gmapping*

Lane Detection

This year, we escape from the paradigm of detecting lanes by fitting lines, and we implement a more computational-efficient lane detection method based on color-segmentation. Since we are following a map-based navigation approach, we consider detected lanes as obstacles in the map. Thus, by obtaining an omnidirectional view of the ground from our *minor* mirror (the small mirror facing the ground in Figure 4), after rectifying the image (applying a function given from calibration), we can simply re-project the blobs of white-like pixels onto the local map as occupied space.

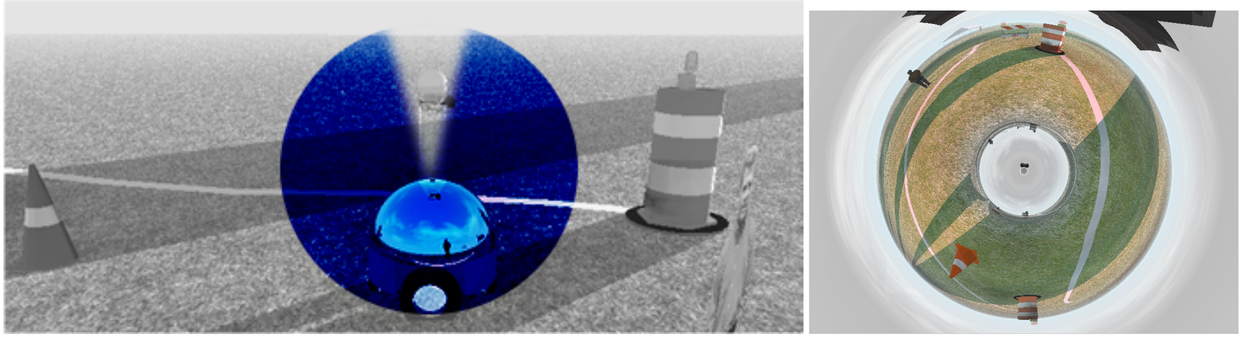


Figure 4. (a) *minor* mirror mostly images the ground. (b) A synthetic view of the imaged ground.

Our technique can cope with curves and broken (dashed) lanes, because knowing the dimensions of our robot *a priory*; it is obvious that crossing a tight obstacle is impossible.

Obstacle Detection

Laser Range Finder

The Hokuyo UTM-30LX is a scanning laser range finder with an extended sensing range out to 30 meters. Scanning rate is 25 milliseconds across a 270 degree range. Due to its longer range and fast response, this range finder is suitable for our outdoor robot moving at high speeds.

Stereo Vision

The Videre© STOC camera (9cm baseline) produces fully synchronized stereo image pairs. These stereo images are then used to obtain 3D point-cloud data. Using Point Cloud Library, the dense point-cloud is down-sampled and outliers removed, after which the ground-plane is extracted. The point-clouds that lie above the segmented ground-plane are used for obstacle detection. To do this, keypoint (feature in an image) matches in corresponding image are performed. Also Sparse Bundle Adjustment library is used for point-cloud matches in subsequent frames giving the position estimates of these obstacles. Consequently, only those 3D point-clouds that serve as obstacles are published as sensor messages in RO. These are then passed to the *navigation* stack for maneuvering. Figure 5 shows the point- cloud data from the Videre camera. The red dots are the obstacles detected in the environment. Obviously, the points in the ceiling will not be present in the outdoor environment. Currently, the stereo camera's horopter has been calibrated to detect obstacles within 4 meters range.

Figure 5. Point -cloud from videre camera

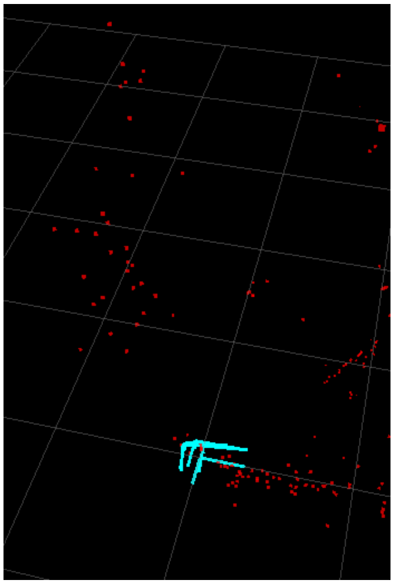


Figure 6. Robot's pose

As well as publishing 3D point-cloud data, stereo vision is also used to localize the robot. Using the *Visual SLAM* (***vslam***) library, the robot's position is mapped out in a new environment. It works by tracking image features between camera frames, and determining the robot's pose and the position of those features in the world based on their relative movement. This tracking provides an additional odometry source, which is useful for determining the location of a robot. This is used to complement laser-based localization methods.

Figure 6 shows the robot's pose (represented by teal arrows) as it moves around and the red dots are the obstacles around the robot.

GPS Assisted Navigation (Outdoors only)

The primary function of the GPS is to localize the robot in its environment implemented as a map. The robot needs to know its position as it navigates around. The robot will be able to identify its position with respect to each waypoints in the track from the data received from the GPS. The robot is then able to calculate its goal position from its current position and move accordingly to reach the goal avoiding obstacles in its way.

The Novatel GPS Device

CATA takes advantage of the Novatel GPS-702L unit to acquire its global positioning information in real time. The information provided by the Novatel GPS is processed in a multi-application architecture which simultaneously feeds the CATA navigation stack and broadcast the robot's global position to external devices. The Novatel GPS Device integrates two independent devices: the Novatel GPS-702L antenna and the Novatel Propak V3 receiver.

The Novatel GPS-702L® Antenna is a dual frequency antenna that delivers excellent performance, accuracy, multipath projection, and L-band functionality. Its robust design provides properties such as waterproofing, durability and bump resistance, which makes this device an excellent choice for mobile robots.

The ProPak-V3 enclosure offers L1 and L2 GPS-only or GPS+GLONASS positions and measurements in real time. The ability to track the satellites of the GLONASS constellation grants greater precision in the positioning solution, and provides industries that rely on GNSS technology increased productivity.

The GPS Controlling Software

Novatel does not provide drivers for its GPS products that are both simultaneously compatible with Linux and ROS. To achieve the integration of this GPS device and the CATA robot we designed the following applications.

The cata_novatel_driver: a node responsible for pulling data from the Novate Propack v3 receiver and broadcast it to the different applications that use such information. The application was designed by our team on C++ using the MobileRobots® Aria library to establish a direct communication with the Novatel Propak v3 receiver using serial port or USB to serial adapter. Once this communication has been secured, the necessary initialization codes are sent to the Novatel receiver, which begins feeding the raw positioning information. The Cata_NovatelDriver application parses the raw positioning information and constructs a XML formatted string containing all the relevant positioning information, which is broadcasted by the same application using the UDP.

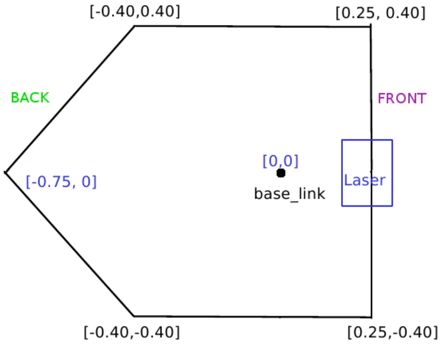
The cata_ros_novatel Application: responsible for the translation of the positioning information sent by the cata_novatel_driver (formatted as an XML string) into a gps_common/fix structure (used by the ROS navigation stack) and the publishing of this structure as a ROS topic.

Path Planning from GPS Waypoints and Map

Path planning in 2-D is achieved by the *costmap_2d* node, which interacts with the *move_base* node and *gmapping* as map server. Thus, we are attempting localization-based navigation (map or model-based navigation) using belief pose-estimation. The 2-D map is constructed on-the-fly via *gmapping*. The current robot's location are filtered (fused) with messages from odometry, compass (IMU) and GPS through an Extended Kalman Filter (EKF) using the *robot_pose_ekf* node. Then, cognition (path-planning and obstacle avoidance) toward a goal (a GPS waypoint) is executed based the current pose estimation (location on the map using GPS coordinates in the *costmap*. The robot receives the messages to move from one location to another. It reads a goal co-ordinate position and then calculates the trajectory to take from its current position by sending velocity commands to the motor controller.

Costmap: Using the following 2-D robot model dimensions of the figure on the right, the *costmap* navigation node generates an inflated map (with an extra 5cm padding) and thus, the path is planned as a trajectory towards a defined goal while avoiding obstacles. There are two levels of mapping:

Local map: reaction to immediate obstacles
Global map: localization and path planning towards goal.



In ROS, obstacle avoidance (course negotiation) algorithms are implemented through the *move_base* node. Whenever an obstacle is found, a new path is recomputed. However, the ***recovery behaviors*** (Figure 7) must be executed first to re-scan the area. From experimentation and simulation, **traps** typically result in the absence of gaps in the frontal direction. Even though the omnidirectional camera can see 360°, the following rotational behaviors are performed to build a robust knowledge of obstacles. during navigation.

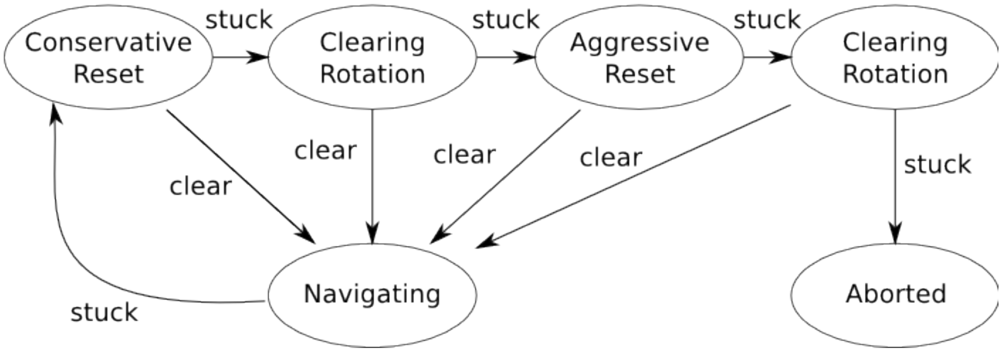


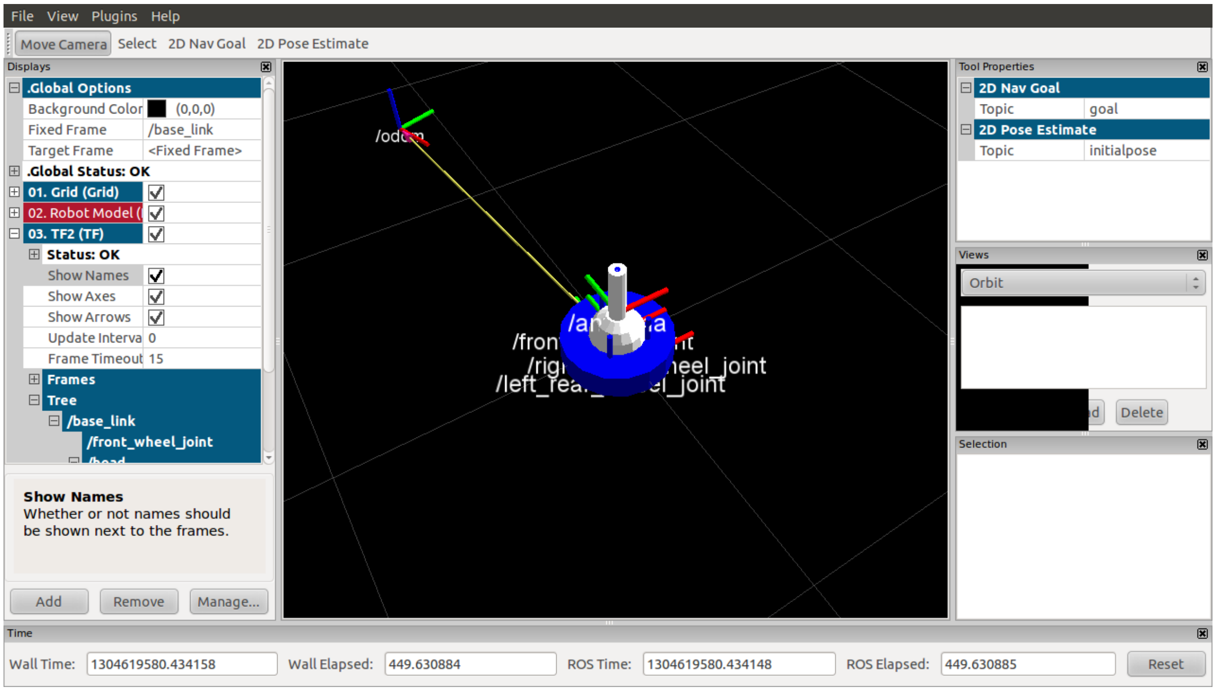
Figure 7. *move_base* node recovery behaviors

Flag Path Detection

In order to distinguish the "green" flags from the grass, we are using color blob finding techniques (existing in ***OpenCV***) on the RGB information from the 3-D point cloud captured by the stereo camera. Since flags are located above the ground, this helps the robot detect "actual height" color blobs which are indicators of flags in the terrain. The path is then re-planned.

Visualization

ROS has a powerful tool for visualization called ***rviz***. It can be used to manipulate robot and also for simulation purposes. Figure 8 shows a simplified 3-D model of the CATA robot using the Unified Robot Description Format (URDF) in the simulation environment with *rviz*. An accurate URDF is important since the transformation between frames (such as sensors, base, and odometry) are directly determined in ROS with the model.



.Global Options
Background Color (0,0,0)
Fixed Frame /base_link
Target Frame <Fixed Frame>
.Global Status: OK
01. Grid (Grid) ☑
02. Robot Model (☑
03. TF2 (TF) ☑
Status: OK
Show Names ☑
Show Axes ☑
Show Arrows ☑
Update Interva 0
Frame Timeout 15
Frames
Tree
/base_link
/front_wheel_joint

Show Names
Whether or not names should be shown next to the frames.

2D Nav Goal
Topic goal
2D Pose Estimate
Topic initialpose

Orbit

Wall Time: 1304619580.434158
Wall Elapsed: 449.630884
ROS Time: 1304619580.434148
ROS Elapsed: 449.630885

Figure 8. CATA simulated URDF model in rviz

Voice Feedback

The CATA artificial voice interface software (*cata_voice*) was designed as a mechanism to deliver instantaneous system notifications using an artificial voice engine. This application allows CATA to deliver spoken messages and notifications in real time without the need of a client computer connected to CATA. This feature was implemented as an improvement in CATA's alerting methods, such as light indicators, especially for real time communication during field operation.

cata_voice was written in C++using ROS and the University of Edinburgh's *Festival Speech Synthesis System*. ***Festival*** offers a full text to speech system with various APIs, as well an environment for development and research of speech synthesis techniques.
The integration of *cata_voice* to the ROS environment makes this service available to all applications running on CATA and compatible with ROS. Different applications can originate voice notifications by publishing messages to the *cata_voice* topic in the form of text strings using ROS libraries.

IV. TEAM COLLABORATION

Collaboration Tools

Although our team is small, maintaining a variety of communication channels to track our progress has been essential for the consistent development of this project. In sum, the main collaboration tools for the team are:

Electronic Mail: 24/7 communication via e-mail and instant messenger.

Project Wiki: A private wiki was setup in order to document the evolution of the system and keep track of our progress and pending tasks. We will make this wiki publicly available after the competition at http://robotics.ccny.cuny.edu/wiki/IGVC2011

Dropbox: File online sharing for manuals and specialized files such as CAD and offline data.

Git: Our in-house version system repository for software development was centralized in our robotics lab server. This code repository will be made publicly available at http://robotics.ccny.cuny.edu/git/ccny-ros-pkg/cata_ros_pkg.git after the competition.

GoogleDocs: To write and share documents, such as this report, our inventory and our budget.

Cost Breakdown

Hardware Components

(No declination value considered)

Part	Manufacturer	Model	Price
Computer	Panasonic	Tougbook CF-30	$5,600
GPS	Novatel	GPS-702L	$8,000
Laser range finder	Hokuyo	UTM-30LX	$5,600
Compass	Ocean Server Tech	OS5000-US	$250
HD Web Camera	Logitech	Pro-9000	$69
DC Motors	Bison	348 Series	$1,050
Brushed DC Motor Controller	RoboteQ	AX2850	$650
Stereo Camera	SRI Videre	STOC-9CM-C-MINI	$1,200
Wireless e-stop	Quasar Electronics	3157	$50

Wheel Hub		262035	$26
Tires and Rims		262206	$60
Analog Panel Voltmeter	Simpson	09780	$110
Analog Panel Ammeter	Simpson	04391	$100
Large Mirror dome			$180
Cast Acryllic Tube (transp)			$250
Construction materials (metal)			$1,900
Construction materials (tools)			$1,600
Misc. Electronic parts			$450
		TOTAL	**$27,145**

Number of Man-hours

We estimate the hours by calculating the average number of hours worked per week and multiplying them by the period of involvement of the project (14 weeks).

Name	Major	Degree Level	Concentration	Weekly Hours
Carlos Jaramillo*	Computer Science	Graduate (Masters)	Vision/Path planning/Hardware	35
Achyut Shrestha	Computer Science	Graduate (Masters)	Vision	25
Sebastian Pendola	Comp. Eng.	Undergrad (Senior)	GPS Navigation	25
Carlos Chinchilla	Comp. Eng.	Undergrad (Sophomore)	Electronics	30
			Subtotal:	**115**

Total Number of Man-hours = 15*115 = 1,725 man-hrs

V. CONCLUSION

We believe that the major improvements applied to CATA (both hardware and software architectures) will provide technical advantage to compete in the upcoming IGVC event. The project was started late in February of this year, but thanks to the new software framework (ROS) in place, we think CATA is well equipped to perform autonomous navigation outdoors. Safety and simplified usability has been our concerned at the moment. In days to come, we plan to follow an exhaustive testing routine as to resolve possible flaws in this new design.