Georgia Institute of Technology
RoboJackets
2007 IGVC Design Report

Table of Contents

1. Introduction 1
2. Team Organization 1
3. Vehicle Design 2
    3.1 Mechanical 2
    3.2 Electrical 3
        3.2.1 Environmental Data Acquisition System 5
        3.2.2 Mapping and Path-finding System 5
        3.2.3 Positioning System 5
        3.2.4 Motor System 5
        3.2.5 Power and Physical Installation 6
        3.2.6 Emergency Stop 6
    3.3 Systems Integration and Software Strategy 7
        3.3.1 Systems Integration Methods 7
        3.3.2 Software Strategy 9
            3.3.2.1 Obstacle and Line Detection 9
            3.3.2.2 Navigation 10
            3.3.2.3 Data Acquisition 12
4. Predicted Performance 13
5. Cost of Project 14

1. Introduction

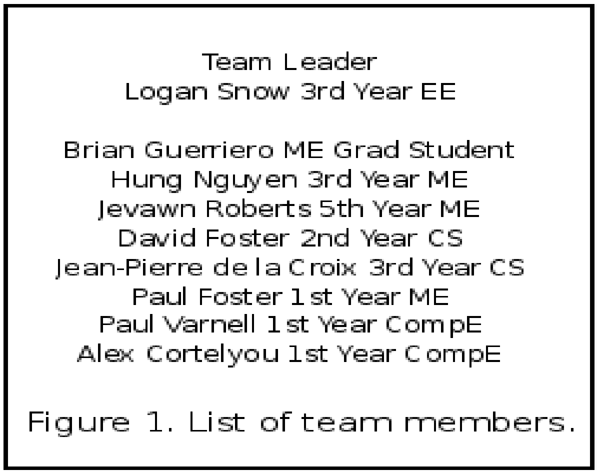
Despite a tremendous effort on the part of three students during the 2005-2006 Intelligent Ground Vehicle Competition (IGVC), the Georgia Tech RoboJackets suffered a significant defeat when mechanical troubles forced us to forfeit most of the competition. However, the lessons learned and experience acquired by the members who participated in the 2005-2006 IGVC have since proved invaluable. Thus, RoboJackets presents Candi, the embodiment of a new year of learning, experimentation, and determination. Candi features sweeping improvements in the areas where last year's submission struggled: robust mechanical design, powerful high level electronics, goal-oriented intelligence, and short-term position state measurement.

2. Team Organization

The team's organization is structured as a recreational competition club for volunteers. The team leader handles general administrative tasks, makes major decisions, represents the group to outside organizations, specifies which tasks need to be completed, and ensures that each active member is able to complete his or her task. All other members are split into two teams – mechanical and electrical/programming. The mechanical team meets two to three times per week during the entire build season to fabricate an entirely new vehicle. The electrical/programming team also meets several times per week to design the new systems and collaborate on code. Within each group they are free to pick any task that they choose on a first-come-first-serve basis.

Three methods keep the members in collaboration with each other: the team leader frequently checks the individual progress of each member and offers help if a difficulty exists, two group meetings are held a week in which the overall project is discussed to ensure continuity, and members are able to update their progress on the RoboJackets wiki. Since ultimate responsibility for completion of a task lies with the team leader, he or she must make certain that work on a crucial task does not stall. If work declines on a task, the team leader has the responsibility of helping the member, getting help for the member, reassigning the task to someone else. Despite the team leader's oversight, most of the key decisions regarding design are made democratically with weight from members who are more familiar with the design concepts in question. In addition to group decisions, use of the wiki also allows members to get involved in others' tasks and contribute, documenting what they accomplish.

Figure 1 shows a list of all RoboJackets members who worked on the IGVC project.



Team Leader
Logan Snow 3rd Year EE

Brian Guerriero ME Grad Student
Hung Nguyen 3rd Year ME
Jevawn Roberts 5th Year ME
David Foster 2nd Year CS
Jean-Pierre de la Croix 3rd Year CS
Paul Foster 1st Year ME
Paul Varnell 1st Year CompE
Alex Cortelyou 1st Year CompE

Figure 1. List of team members.

Approximately 2280 man hours were spent designing, building, and testing the Candi system. This time period includes the fall, spring, and first month of the summer semesters during the 2006-2007 school year.

3. *Vehicle Design*

The first step in the design process was to determine the weaknesses of the previous year's submission and to think of preventative measures for implementation in Candi. The motors of the previous submission had not been properly geared, resulting in the robot being stalled on grass. In order to get a true assessment of the robot's abilities, properly geared motors were installed. Several test runs were conducted using white side walk and orange barrels to simulate the IGVC course. A reactive avoidance algorithm had been developed to create paths that steered the robot away from obstacles. While this algorithm proved successful for pure obstacle avoidance during the tests, having the robot progress forward on a path was a slow and random process. Thus, building a robust mechanical drive base and a sense of intelligence about location and paths became primary goals for the 2006-2007 IGVC submission.

3.1 Mechanical

The mechanical team opted to design and fabricate an entirely new chassis and drive train. Similar to the robot used last year, the new design, Candi, utilizes a differential drive system. Differential drive was preferred over other systems such as “swerve steering” and “crab drive” because it would prove easier and more efficient to program and control.

The new custom frame was fabricated from thin wall one inch square steel tubing, tapering in the rear. Each piece was cut and welded from six foot lengths of stock. Aluminum bars were used to mount the drive motors.

The two differential drive motors were donated from another robotics team. They are NPC Robotics 1.7 hp 24V DC motors. These motors power the front two drive wheels, moving the entire vehicle. The traditional system of two swivel casters supporting the rear of the vehicle was avoided for several reasons:

1. Swivel casters, from past experience, have introduced significant amounts of wobble into the rear end of the vehicle while traversing through soft grass, making control and vision more difficult.
2. Zero radius turning with traditional casters induces slight forward or backward translation to the entire chassis as the wheel swivels around its caster angle axis, introducing uncompensated motion to the controllers.
3. To maintain a ground clearance of approximately 6-7 inches, casters of only 3-4" diameter would be required, much smaller than those desired for traversing grassy terrain.

The mechanical team decided to abolish the standard practice of utilizing swivel casters and implement one single 12" ball caster. This omni-directional ball caster solves the wobble issues, and can instantaneously compensate for changes in direction due to its zero degree caster angle. The 12" diameter ball will also leave a wider footprint in soft grass. The stainless steel ball is held in place by a spring steel "net" of five small ball casters and four small roller bearings around the equator.

Since the main vision system camera was desired to be mounted as high as possible for the optimum field of view, chassis stability was placed as a top priority. To achieve stability, an independent suspension system was implemented on the front drive wheels. The inverted cantilever style suspension keeps the drive wheels behind the main pivot, effectively shortening the wheelbase, allowing for tighter turning. Suspension is provided through the implementation of two single acting pneumatic cylinders with coilover springs. Small valves were installed at each cylinder port and adjusted to introduce damping. Since the 5' camera post is mounted to the rear of the vehicle, the main rear ball caster base-plate was also suspended by eight small springs to allow for shock absorption.

3.2 Electrical

In order to achieve powerful multi-tasking system intelligence, the electrical/programming team decided to focus on systems with operating systems as opposed to lightweight microcontrollers. Through brainstorming and study of other systems, the team realized that the new system would need four major components: a system to acquire data from the environment and create a map, a system to find a goal point in the map and an optimal path to the point, a system to provide feedback describing the system's coordinates and heading, and a system to turn path commands into locomotion.

By distributing these tasks on different systems, a more robust and intelligent system can be created. Specific benefits include: systems that need to run at different speeds for optimal processing may do so; systems are able to provide more processing power to their specific applications; systems are less susceptible to complete crashes; and systems are easier to develop in a modular method with collaboration on their interfaces. Figure 2 shows the distribution and connection of all proposed systems.

As shown in Figure 2, all systems are connected via a wide bandwidth Ethernet connection. In the software section, the system's powerful robot server that allows for easy distributed

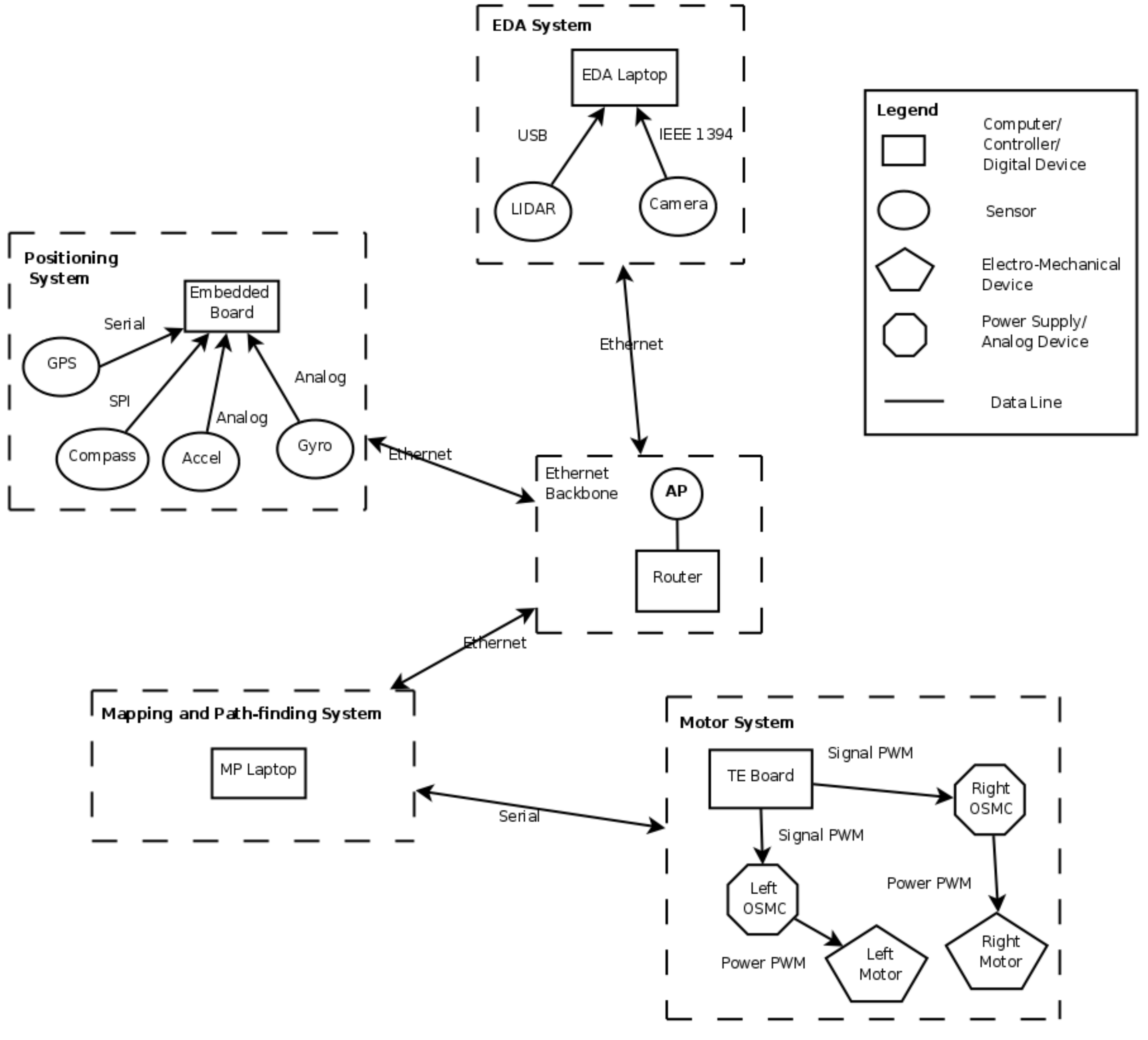


Figure 2. Candi systems diagram.

computing will be discussed.

3.2.1 Environmental Data Acquisition System

The Environmental Data Acquisition System (EDA) constitutes the first system in Candi's control loop that provides input from the environment. The EDA is centered on a laptop built upon an Intel Core 2 CPU clocked at 2 GHz and running Ubuntu Linux. With 1 GB of DDR RAM, a 74 GB hard disk, and an NVIDIA GPU, the laptop is powerful enough for real time computer vision processing as well as creation of data maps. For capturing data regarding lines and barrels, a JVC consumer-grade camcorder is connected to the EDA via IEEE 1394. A camcorder was chosen since it satisfies three criteria: it has a low cost, can adapt easily to changes in lighting conditions, and has high data output. However, to allow for limited use of computer vision, a Hokuyo URG LIDAR is connected via serial to detect general obstacles.

3.2.2 Mapping and Path-finding System

The second requirement of a full control loop is satisfied by the Mapping and Path-finding System. This system is implemented on a laptop with the following specifications: a Pentium IV 2.4 GHz CPU, 512Mb of DDR RAM, and a 60 GB hard disk. Like the EDA laptop, this laptop also runs Ubuntu Linux. The laptop acts a robot controller, taking data from the EDA, creating simple maps, selecting goal points, and determining paths.

3.2.3 Positioning System

Basic positional feedback is provided to the Mapping and Path-finding System by the Positioning System. The system's central computer is a 200 MHz embedded computer from Technologic Systems based on the ARM9 processor. With the ability to boot Debian Linux kernel 2.4, the embedded computer is able to communicate with the larger systems via Ethernet while managing multiple processes at low monetary cost. Four sensors are attached to the embedded computer: a SiRF III GPS Evaluation Board connected via serial, a Vector 2xe 2 axis compass connected via SPI, and a 3 axis accelerometer and 2 axis gyro connected via ADC's. The GPS provides waypoints for the Navigation Challenge, the compass helps keep Candi on track during the Autonomous Challenge, and the accelerometer and gyro provide local position information for a limited distance.

3.2.4 Motor System

Once a direction of travel has been determined by the Mapping and Path-finding System, data is sent to the Motor System to provide locomotion. The electrical/programming team inherited a small control board based on the AVR ATTINY8 MCU and the Open Source Motor Controllers (OSMC) from other groups within the RoboJackets. In order to reserve funds for other components, the control board was turned into a PWM generator that can be connected to the Mapping and Path-finding laptop via a serial connection. Then, interface circuitry was created to connect the control board to the OSMC's over an optically-isolated connection. This powerful system was created to help ensure that motor difficulties would not be a problem in this year's submission.

3.2.5 Power and Physical Installation

The electrical system is divided into two completely isolated systems. One 24V system powers the main drive motors, and one 24V system powers the computers and sensors. The two systems are separated by opto-isolators sending PWM data to the motor drivers. The drive motor system is powered by two large 12V lead acid batteries grounded to the chassis. The batteries are mounted on heavy drawer slides on one side of the vehicle to allow for easy access and charging. The two small 12V motorcycle batteries powering the computers and sensors are strapped into an internal tray on the opposite side of the vehicle.

A custom-designed power supply provides 20, 19, 12, -12, 5, and 11V to provide a maximum of 22A to the logic systems. The power supplies were designed utilizing analog switching power IC's from National Semiconductor. Due to careful component selection, parameter tweaking, and board layout, the power supplies can achieve efficiencies greater than 90%.

Important circuit boards were mounted in individual aluminum boxes. These boxes were then mounted to the chassis. The motor driver boards, emergency stop system, embedded board, and power supplies are all individually boxed and fastened to the vehicle frame.

3.2.6 Emergency Stop

In order to protect against system anomalies, an emergency stop system sits between the motors and the batteries. This emergency stop is both simple and effective. A large power relay is configured as a latching relay with a large red button and the relay of a wireless garage door opener in its signal path. Initially, a pushbutton must be depressed to energize the power relay into latching mode. If either the red button is depressed or the wireless transmitter is enabled, the latching power relay's signal line will open, cutting power to the motors.

3.3 Systems Integration and Software Strategy

3.3.1 Systems Integration Methods

The spinal cord of Candi's system is the open source robot server Player. This server, developed by the open source community and by academia, abstracts multi-threaded processing, exact system signal control, and Ethernet transfer between computers. The Player server offers a powerful, easy, and innovative approach to systems integration. On each computer with an operating system, Player runs as a server providing device interfaces. Device interfaces are implemented as plugins to the Player server. For example, on the EDA Player runs a driver for the Hokuyo URG LIDAR. Processes running on any of Candi's computers may address the LIDAR and draw data from it as if the LIDAR were running on their respective systems. With a built-in queue implemented on each server, Player can handle traffic from a large number of processes trying to communicate with devices to which they have subscribed.

To simplify communication between computers and to allow device drivers for Candi's different hardware components to be developed separately, the electrical/programming team decided to write much of the code as plugins for the Player robot server. In addition, the Player distribution already had a standard driver written for selected LIDAR, so time saved in writing that device driver. However, Player only had a standard camera driver for cameras that are compatible with the raw1394 library. Unfortunately, the camera did not work with the raw1394 library, although it was compatible with the dv1394 library. Thus, a custom dv1394-based camera driver was written in order to talk to the camera using the IEC 61883-2 DV (Digital Video) protocol. These drivers were set to run on the EDA Player server.

On the embedded board Player server, the ability to communicate with the GPS, compass, accelerometer, and gyro is abstracted as a device driver plugin. Other systems may query the device plugin “IMU” for the following information: global position in latitude and longitude coordinates cardinal direction in degrees, and current position relative to a starting point. By the time of the competition, some of this data will likely have changed due to removed or added features.

The central “client” process runs on the Mapping and Path-finding System along with a Player driver plugin that allows communication with the motor driver controller. This process can be thought of as the “main” programming loop in the system; it queries plugins on the EDA and the Positioning System for data when it is ready to determine the next direction of travel.

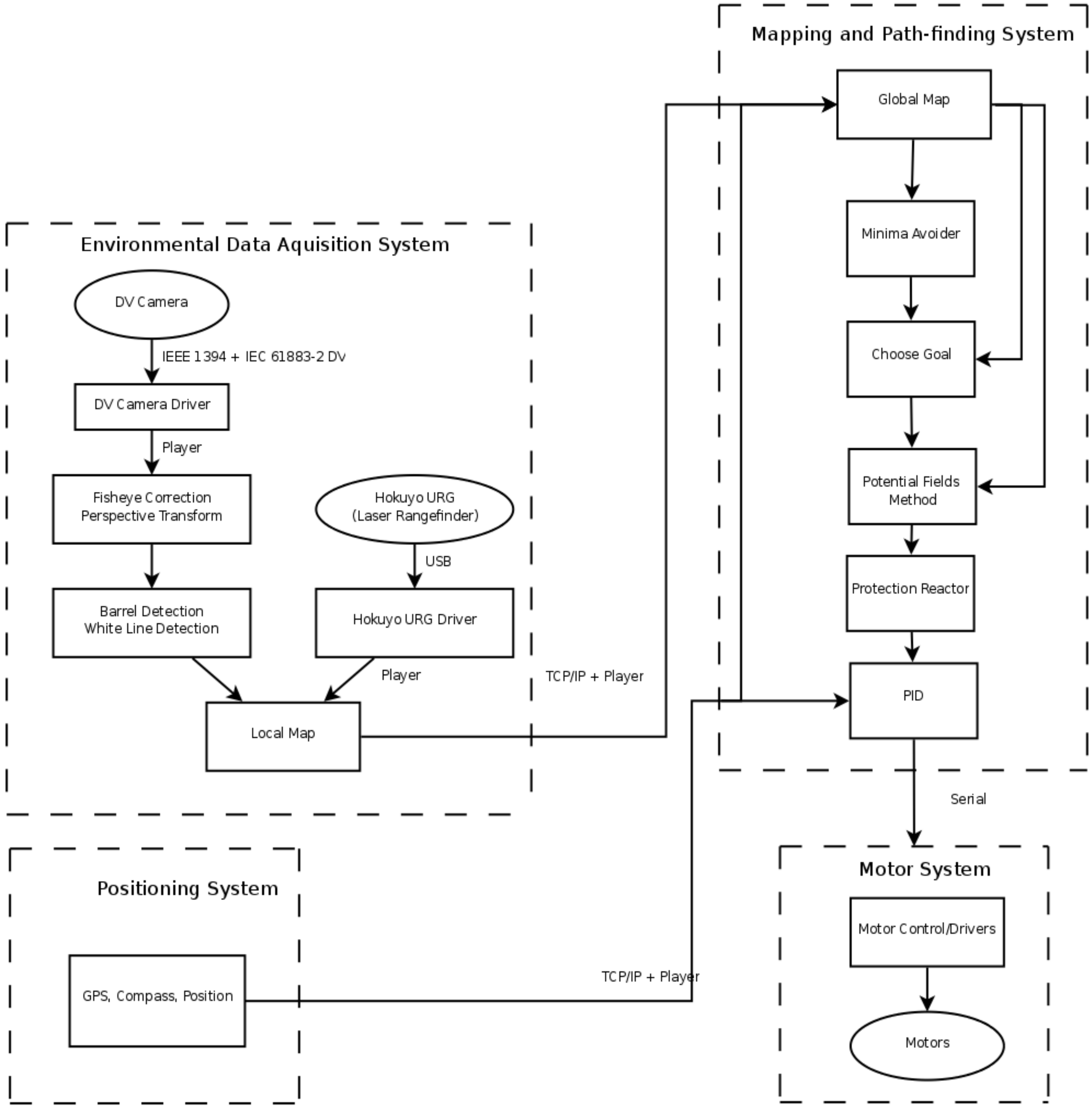


Figure 3. Process diagram showing all of Candi's systems.

Figure 3 shows the complete, integrated process diagram.

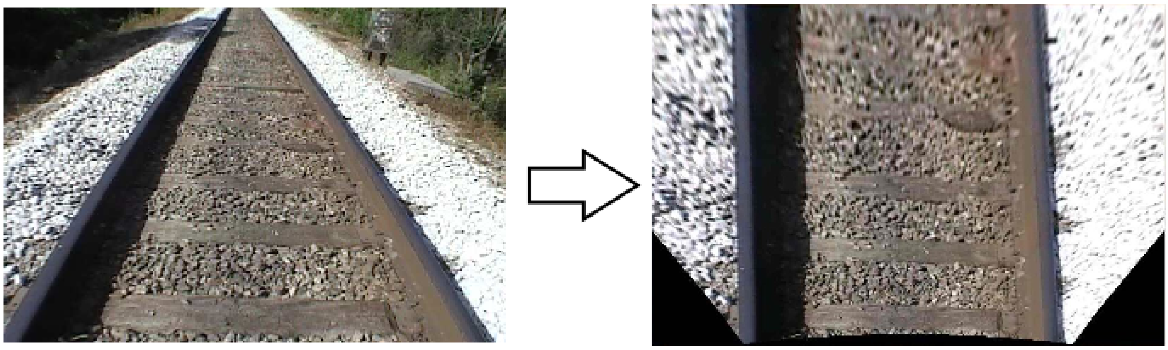
3.3.2 Software Strategy

3.3.2.1 Obstacle and Line Detection

Data on white lines and orange barrels (for a sanity check with the LIDAR barrel data) is provided by a powerful implementation of the perspective transform followed by white blob and barrel identification and the Hokuyo URG LIDAR. The goal of the perspective transform is to reverse all of the changes that occurred to an image from the time light leaves the ground to when it is detected by the camera. Once the transform has run, all of the points in an image have been correlated to match their real Cartesian coordinates away from the camera. Any points in the transformed image that appear as white lines are added to an environment map; additionally, detected barrels are checked against LIDAR data to ensure that an agreement exists between the two methods. If so, then LIDAR obstacle data is placed into the same map as the camera line data. This method ensures robust, reliable data acquisition from the environment.

In designing the perspective transform, many points had to be considered. Two types of distortion had to be considered – perspective distortion and lens distortion. These distortions had to be inverted in the reserves order of their mentioning. While most camera lenses approximate that of a pinhole camera (a gnomical lens), an equidistant full frame fisheye lens was used on the camera as it has the advantage of wide angle of view without the high monetary cost of a wide angle gnomical lens. This lens caused barrel distortion, where rather than the magnification of an image being constant throughout, it increases with radial distance from the center of the image. By measuring several lens parameters, pincushion distortion was introduced to exactly negate the barrel distortion.

Next, the perspective distortion was negated to “transform” image coordinates to real coordinates. First, the matrix calculations normally used to produce perspective distortion (giving the appearance of perspective) in virtual 3D environments were mathematically inverted. Next, a factorization of the matrix was coupled with was used to couple variables created by a lack of depth information in the image to those created by a lack of height information from the ground to produce a transformation matrix that correlated to the perspective distortion created by the camera lens. An example of the perspective correction is shown is Figure 4.



Uncorrected Image

Perspective Corrected Image

Figure 4. Demonstration of the perspective transform correction.

The perspective transform was designed to take advantage of several features offered by OpenGL, the open source graphics library. Since one of the primary functions of graphics cards is performing perspective transforms, utilizing OpenGL allowed the transform to be processed completely by the GPU of the EDA laptop. Also, data can be dumped directly from the GPU into RAM, leaving the CPU free to run the Player server and to process maps and other algorithms. By using the GPU, the lens and perspective corrections were able to process approximately 50 frames -per-second in real time.

Since proven legacy code existed for the detection of barrels and white lines, the software was ported to work with the new system. In order to detect barrels for correlation with LIDAR data, a red green blue (RGB) color space containing data of interest was loaded. Then, the green sample of the color space was subtracted from the red sample. The resulting data represented a measure of the orange intensity of any given subset (pixel) of the color space. Since IGVC barrels were determined to be of a very high orange intensity, pixels were then compared against an experimentally-determined threshold. Pixels that had orange intensity values greater than the designated threshold were then passed to the barrel detection algorithm; pixels failing to meet this criterion were discarded.

White blob detection required more processing to ensure accurate detection. White was determined to be too difficult to process by only thresholding high intensity values in the RGB color space. From experimentation with the camera, many high intensity subsets of the image space were found to appear close to white. In order to solve this problem, conversion to a hue saturation brightness (HSB) color space before processing was proposed. By converting the input image to HSB, intensity values of saturation (purity of the color) and brightness (intensity of the color) could be assessed. The measure of saturation was especially important, as a great deal of the uninteresting white data in images appeared somewhat grayed by mixing with other intense colors. Pixels values with saturation less than a certain threshold and brightness above a certain threshold were then considered line or pothole objects.

3.3.2.2 Navigation

The navigation is based around the Potential Fields Method (PFM). In the PFM, obstacles in front of the Candi generate a repulsive force and the goal generates an attractive force. Candi moves in the direction of the sum of these forces, which should lead it towards the goal while avoiding obstacles locally. The PFM has the advantage of being one of the easiest navigation methods to implement and is more than fast enough to run in real time. Motor speeds are also generated directly, which would have to be calculated and corrected when using a method that just generates the path to the goal (graph searches).

PFM's primary limitation is that it cannot, in some situations, lead Candi to its goal by itself; it can be trapped in places where the repulsive forces of the obstacles exactly balance out the attractive force of the goal. At these places the potential is at a local minimum. This innovative implementation attempts to adjust the parameters of the algorithm to prevent these situations from occurring and avoid them if they do occur.

To use the PFM, Candi must know the location of obstacles and the goal relative to itself. The locations of obstacles are passed to the navigation system from the vision system. For the Navigation Challenge, the goal is taken to be the one of the GPS waypoints. Once Candi reaches a waypoint, the goal is set to be the next waypoint. The order in which the waypoints are visited are determined by a simple graph search before the beginning of the competition. For the Autonomous Challenge, the goal is not specified in a way that can be immediately used, so it is calculated from information on the map. The goal is calculated to be halfway between the most distant visible portions of the white lines. When only one of the lines is visible, the last good goal is used.

As mentioned above, a significant difficulty with using the PFM is making sure that Candi does not become trapped at places where the potential is at local minimum. The likelihood of these situations appearing can be reduced by careful selection of the parameters associated with the method. Values such as the attractive and repulsive gains are controlled by a genetic algorithm that selects values based on how pronounced any local minima are on a particular map of the environment. While this algorithm could run during the competition, adjusting the parameters at regular intervals, it is unnecessary to do so as long as the system is calibrated with real data beforehand. If any local minima do occur, another system is running in the background to avoid them.

Maps of the environment are checked periodically for any local minima. If a local minimum is present, a local goal is chosen closer to Candi's current position so that it will not be trapped. This selection is processed by preformed a graph search on the map to find a path to the goal, and then placing the new goal to be used by the PFM at a point on the path near Candi. This way the PFM can still be used to generate the motor speed, although it is guided by another path planner. However, the alternate path planner is significantly slower than the PFM, so the path is not rechecked often after it is first generated, which could cause problems if the Positioning System accumulates too much error over time.

After the motor speeds are generated by the navigation system, they are sent to an independent process that performs one final check to make sure that Candi is not in immediate danger of a collision. If an object is detected very close to Candi by the LIDAR, this process stops the robot and waits for the incoming commands to normalize. This protection ensures that even if the vision or the navigation system fails, Candi will not contact an obstacle. As a final check, compass headings are compared to ensure that Candi has not become turned around on the course. If no problems arise, the motor speeds are sent to the on board motor control system.

The motor control system receives motor commands from the other systems. Motor speeds are regulated by Proportional/Integral/Derivative (PID) control and then sent to the Motor System via a serial port. The Motor System then uses pulse-width modulation to control the actual speed of the motors.

3.3.2.3 Data Acquisition

To provide positioning information for navigation and PID control, the Positioning System implements a simple Inertial Measurement Unit (IMU). GPS and compass data are extremely stable; therefore, the system only need poll those sensors for information when the information is required. Also, no digital signal processing is required for those sensors. However, the accelerometer and gyro data must be sampled in real time in order to integrate their results into positional data. Since the Mapping and Path-finding System does not require a continuous stream of local position data, the system enables and disables local positional tracking on the Positioning System when necessary to avoid buildup in long-term error.

The Positioning System is implemented on an embedded Linux computer. Real time processing in operating systems such as Linux is not a trivial task, as many processes may try and run without warning. However, the electrical/programming team has found an innovative method for overcoming this difficulty. By assigning the ADC sampling process the highest possible system priority, locking the process in memory, and synchronizing it with a real time clock, the system can achieve accurate timing at approximately a 2 millisecond granularity. Five ADC's must be sampled in order to capture all of the axes from the accelerometer and gyro, so each channel can be sampled at 100 samples per second. This value, which allows for detection of changes at a rate of up to 50 Hz, is more than adequate for tracking the motion of the relatively slow-moving Candi platform.

4. Predicted Performance

Unfortunately, all systems have not been tested in an integrated manner as of the writing of this design paper. Therefore, speculation may only be provided based on the operation of the individual systems. Candi processes information quickly and has a robust, reliable drivebase; therefore, speed, ramp-climbing, and reaction times should allow Candi to move through the courses at about half the maximum speed. Since two battery systems are being utilized along with efficient power supplies, battery life should allow for at least two hours of constant run-time between battery replacements. Distances should be detected at most about two meters from the LIDAR and three meters from the camera. Due to the intelligence of the navigation software, traps should be escaped eventually with a period of searching. The GPS is accurate to two meters, so Candi should be able at least to touch most of the waypoints.

5. Cost of Project

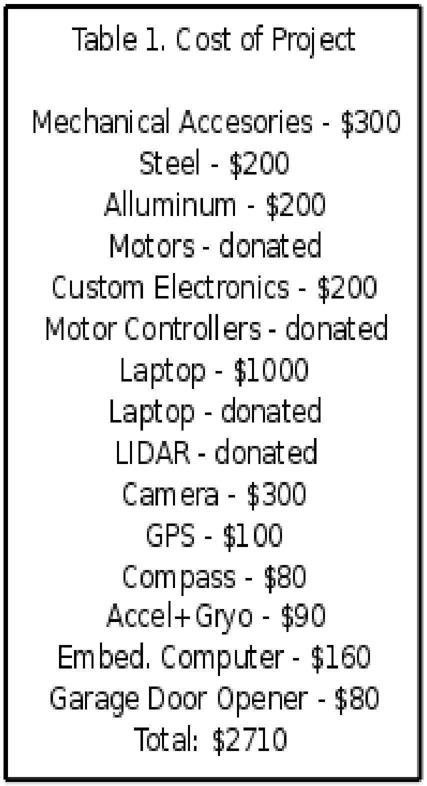


Table 1. Cost of Project

Mechanical Accesories - $300
Steel - $200
Alluminum - $200
Motors - donated
Custom Electronics - $200
Motor Controllers - donated
Laptop - $1000
Laptop - donated
LIDAR - donated
Camera - $300
GPS - $100
Compass - $80
Accel+Gryo - $90
Embed. Computer - $160
Garage Door Opener - $80
Total: $2710