Rose-Hulman Autonomous Terrain Traverser

Michael Auchter, Jay Kinzie, Jon Klein, Tom Most, Andy Spencer
{auchtemm,kinziejh,kleinjt,mosttw,spenceal}@rose-hulman.edu

Robotics Team, CM 5000
Rose-Hulman Institute of Technology
5500 Wabash Avenue
Terre Haute, IN 47803-3999

May 28, 2009

This report and the described vehicle were designed and constructed by the Rose-Hulman Robotics Team and the work done during the 2008-2009 school year is of equivalent caliber to that which would be given credit in a Senior Design course.

_____
David S. Fisher

1 Introduction

The RHIT Robotics Team is comprised of undergraduate students from Rose-Hulman Institute of Technology in Terre Haute, Indiana. We have a long history with the International Aerial Robotics Competition, having built autonomous helicopters for the last fifteen years. However, due to a recent shift in interests it was decided that a ground based competition would be more appropriate for the team's main project and the Intelligent Ground Vehicle Competition was selected as a primary competition.

Our entry this year is called the Rose-Hulman Autonomous Terrain Traverser version 2 — RATT 2. It reflects experience gained with the first revision of RATT, which was unsuccessful at the 2008 competition for a number of reasons.

2 Design Process

2.1 Design Decisions

Use of Free/Open Source components

Based on our previous experience using proprietary protocols and software we made the decision to use Free and Open Source software and hardware wherever possible. The software for the robot consist of entirely free software and is released under the GNU GPLv3. Several electrical components, such as our primary camera, also have the schematics and software available under the GPL.

Modular and expandable design

Our entry this year is intended largely as a platform for further growth, particularly with regard to hardware and electronics. Thus our design is composed of components picked or designed to be reusable in future years. For example, we picked motor and gearbox assemblies with the intention that they be used in multiple chassis designs, so they are durable and servicable by the team. Similarly, we intend to use a different communications interface for our custom-made electronics boards next year, so the boards were designed to accomodate the necessary controller ICs.

Off the shelf components

We seek to balance the need for education, timely completion of the project, and budgetary constraints. Thus, we seek off-the-shelf components whenever they are appropriate. Some examples include our motors and gearboxes, which are more reliable than the custom chain drive we used last year, and our motor controllers, which have proven superior to the custom units that prevented us from competing last year. In our software we try particularly hard to use existing solutions, detailed below.

2.2 Administrative Facilities

2.2.1 Lab space

Lab space and equipment has been provided by the Rose-Hulman Student Government Association.

2.2.2 Computing facilities

The team provides computing facilities for club members to use while working on the project. Access is set up using a LDAP server so that users can log into any services using the same account.

Wiki

The main use of the wiki is for documentation and communication within the team. It also serves as a starting point when new members wish to join the club.

Subversion
Primarily used by the software team, Subversion serves as a source code management system. We chose to use SVN due to the ease of use and familiarity incoming club members have with it.

File storage
A Samba/SSHFS server is used to host large files like team photos and video, as well as to back up the robot's CompactFlash card.

Mailing list
A mailing list is provided by Rose-Hulman Institute of Technology and is used to send meeting agendas and reminders.

IRC
IRC is often used for team communication between official meetings. Logs of the official IRC channel are maintained on the club website. The IRC channel is provided by FreeNode.net.

3 Hardware Platform

3.1 Overview

The mechanical design of RATT 2 reflects the experience of the team with constructing and building a vehicle for the 2008 IGVC. In particular, the previous frame was insufficiently sturdy and difficult to maintain, the drivetrain was unreliable, and the handling characteristics were not ideal for the competition's goals.

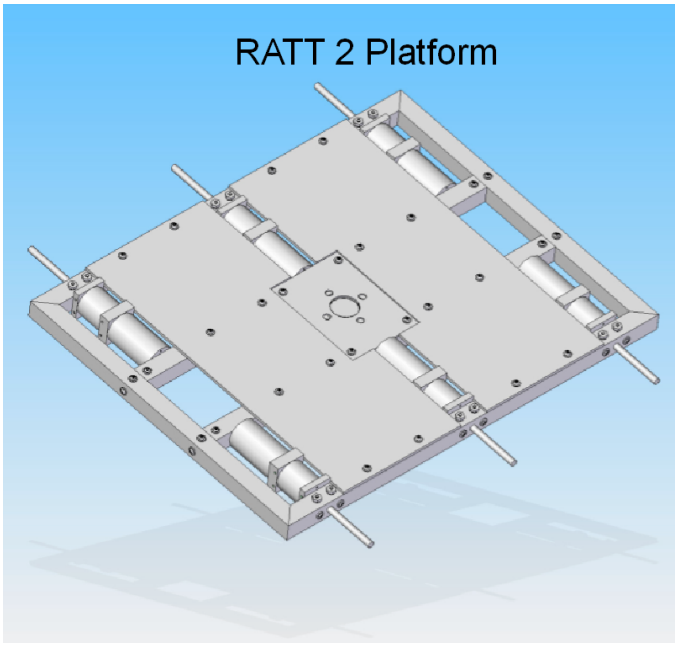

Figure 1: RATT 2 Platform

3.2 Drive Train

The previous design, RATT 1, had two wheel drive and two casters. RATT 1's drive wheels often had problems maintaining contact with the ground. To alleviate this problem RATT 2 has 6 wheels, all of which are driven. Since all the wheels are driven, the robot will have control at all times. In addition, the 6 wheel design will also improve turning characteristics. RATT 1's center of turning was not at the center of the robot. This greatly increased the size of the radius of safety around the robot, hindering navigation around obstacles. To compound the turning issue, the

camera mast is not placed in the rotational center of the robot, making surveying the area around the robot (by rotating the robot in-place) more difficult. RATT 2 fixes these problems by using the 6 wheel drive to turn about is geometric center; thus reducing the radius of safety and proving an ideal location to mount the camera mast. Should it become necessary, the frame has been designed to permit easy addition of a suspension at a later date.

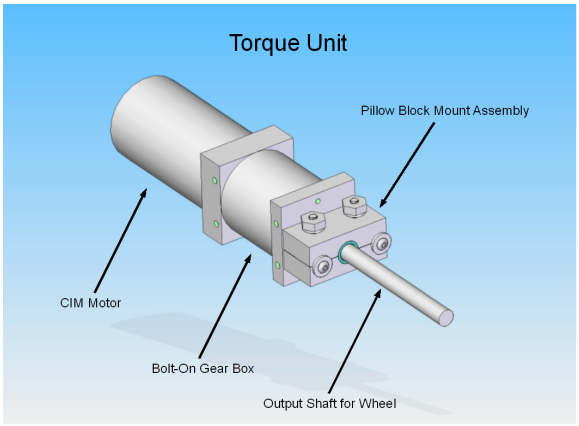

Figure 2: Torque Unit

The motors that RATT 2 uses are the same as RATT 1's — FIRST CIM motors. These motors have many features which make it ideal for the design scenario. The motors are cheap. They cost $28 each and can be purchased online at www.banebots.com. Many of our team members have participated in FIRST and are acquainted with the motor specs and correct usage. Using the same motor also allowed us to have spares mounted on an operational RATT 1, permitting us to use RATT 1 as a test platform while RATT 2 was constructed. The motor also uses an industry standard C-Face mounting scheme. This broadens the available equipment that can be paired with the motor, such as the gearboxes we selected for use on RATT 2. The new gear train is also a great improvement — RATT 2 has been designed to use a 27:1 gearbox that mates directly to the motor. This makes the gear train compact and efficient, and allows the design to use 6 discrete torque units to propel the robot through the course. Each one of the units can be individually serviced in the pit at the competition. In contrast, RATT 1 used a 2 stage chain drive for propulsion. The chain drive dependes heavily on proper chain alignment over a significant distance. When combined with the 80-20 frame, this has led to frequent chain failure. RATT 2's gear boxes use planetary gear sets for reduction. This is a much more reliable way of transmitting the torque RATT 2 needs. In addition, the individual components of the gear boxes are sold individually, and the gear boxes are made to be disassemble-able. This allows easy maintenance and care of the gearboxes.

RATT 2 also uses different drive ratios than RATT 1. RATT 1 is too fast and lacks the torque necessary for precisely controlled movement. RATT 1's 20:1 drive has been replaced with a 27:1 drive. The extra torque will help the movements of the robot be more controlled and deliberate. During testing, RATT 1 was found to be fast enough that the team had to run with the robot moving at full speed. This extra speed was wasted because the maximum speed for the IGVC is 5 mph. The 27:1 drive of RATT 2 will drive the robot at approximately 6 mph maximum, limited to 5 mph in the motor control software.

3.3 Frame Design

RATT 2's frame was designed to be constructed out of easily available materials and builds on previous design and manufacturing experience with RATT 1. The frame is made from 1.5 inch square aluminum tubing with $\frac{1}{8}$ inch thick walls. Rose-Hulman's machine shop has large amounts of this tubing in stock. This makes it a low cost framing solution, permitting us to re-manufacture any defective or broken components. Since the cost of mistakes is not high, new team members are able to get valuable machining experience with an integral part of RATT 2's design, furthering the pedagogical goals of the team. The minimum size of the robot per IGVC rules is 24 by 36 inches. One of our design challenges is to get as close to this minimum size as possible in order to maximize the maneuverability of the robot. RATT 2 is 28 inches wide by 36 inches long — only slightly wider than the minimum (some extra width is necessary

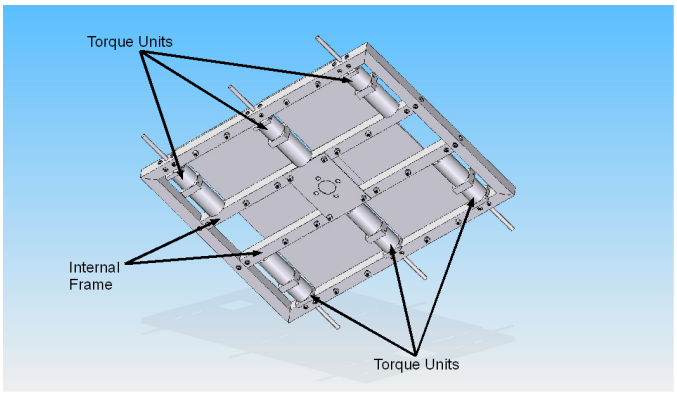

Figure 3: Underside of RATT 2's frame

due to the motor/gearbox pairing, which need to be able to slide out from the inside to permit assembly/disassembly of the frame).

Two frame parts also run laterally along the inside of the robot, forming an internal frame. The purpose of this internal frame is to provide support for all of the robot's electronic and sensory equipment, including the camera mast. It also stabilizes the motors by cradling the extruded lip on the back of the motors, which is intended to reduce the vibration caused by and applied to the motor, prolonging the life of the motor and gearbox.

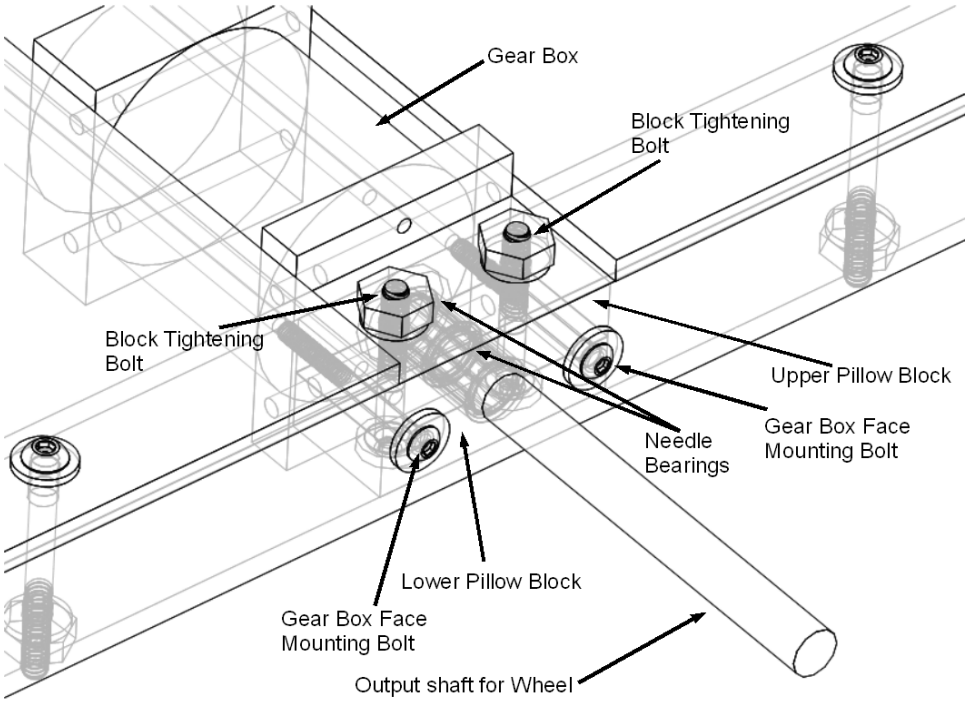

Figure 4: Torque unit assembly closeup

The front of the motors mounts to the frame in a clever way, via pillow blocks that are slid inside the tubing of the outer frame. There is a total of six of these blocks, each containing two needle bearings to support the output drive shaft of the associated motor. The motors are mounted to the frame and pillow blocks via bolts through the frame and blocks that mate with threaded holes in the gear box face. This seals the pillow blocks' bearings inside the frame, prolonging the life of the bearings by keeping dirt and other environmental factors away from them. There is a small

cut out on the inside of the frame that is co-axial with the main shaft hole to permit the replacement of the bearings. As mentioned earlier, the gearbox face seals this cutout in the frame.

3.4 Summary

Building off of last year's experience, the team has produced a design intended to both address the flaws of the current one and better address the demands of the competition. This is done by emphasizing mechanical power and ruggedness, cost and ease of manufacture, and lessons learned with regard to turning radius and physical size.

4 Electrical Systems

4.1 Computer

As image processing is very CPU and memory intensive, the computer on RATT is an AMD Athlon x2 5000+ with 4GB of RAM. This provides for ample computational power for doing both the image processing and other robot control. In particular, the dual-core processor was chosen to allow image processing code to run on one core, and the general control code to run on the other core.

The operating systems and control software are installed on a 2GB Compact Flash card connected through an IDE→CF converter. A solid-state storage media was chosen for two purposes: to reduce the power consumption and eliminate moving parts. The computer and Compact Flash card are powered using an off-the-shelf 160W DC-DC converter.

The computer hardware is identical to that used in the 2008 competition. However, the case for the computer is different. It is mounted in a custom made enclosure with most of RATT's other electrical equipment. This new enclosure provides easier access and better cooling for the computer and other equipment. Nearly all electronics have been centralized in this case.

4.2 Motor Controller

Motor control for RATT is accomplished using two devices: Innovation First Victor 883s for controlling the speed of the motors, and custom designed controllers for providing closed-loop control of the robot's speed.

The Victor 883s were chosen for a number of reasons: they handle the current necessary to drive our motors, they are relatively inexpensive, and they are easy to work with. Additionally, some members of the team had prior experience with them.

The custom designed boards serve to provide accurate speed control of the motors, and to interface the Victors with RATT's main computer. Each board can control up to three Victor 883s, and two boards are used for RATT: one to control the left side, and one to control the right side.

A Grayhill 63R, 256PPR optical encoder connected to the center wheel's shaft on each side of the motor provides feedback to the controller board. The encoder's output is fed into a Proportional-Integral-Derivative closed-loop controller which runs on the controller board's microcontroller.

The controller board communicates with the computer over RS232; all of the settings on the controller board are able to be changed over this interface, including the desired speed and the three gains for the PID loop. A number of useful statistics are also able to be reported over this interface, including the wheel's current speed and the current values for the P, I, and D components of the algorithm.

4.3 Wireless Emergency Stop

The emergency stop was designed to fulfill contest requirements of a remote kill switch. An Atmel microcontroller on the robot and on a battery powered hand-held controller communicate using an off the shelf wireless Zigbee

transceiver. If the connection between the two devices is lost, or if the power is lost to the emergency stop board, power will be cut to the motors and the robot will be unable to move. If the transmitter is activated, power will be cut to the motors.

4.4 Accelerometer

The accelerometer on RATT is a MicroStrain 3DM-G which interfaces with the computer over RS-232. This accelerometer has a three axis angular accelerometer, as well as a three axis linear accelerometer and magnetometer. This allows the accurate calculation of current heading and position relative to the starting point by integration. This device is primarily used as a compass and to discover the camera's orientation for rectification of the ground plane.

4.5 Camera

The camera used for line-detection and obstacle avoidance is an Elphel 353. This camera was chosen for a number of reasons: it allows for user-selectable frame-rate and resolution, it is accessible over Ethernet, and the hardware and software are licensed under the GPLv3. This last point was especially important: since the source code for the camera's onboard FPGA was available, we have the option to perform image processing directly on the camera itself.

4.6 Power Distribution

Improving the power distribution was one of the focuses of the electronics team this year. The design for last year did not always use polarized connectors where appropriate and time was lost frying boards. This year, circuit breakers, polarized connectors, bus bars, and fuses were used extensively.

The power system of the robot consists of two separate 12V, 44Ah sealed lead-acid batteries. One of these is dedicated to powering RATT's six motors, which account for the majority of the robot's power consumption, and the other is used to power the computer and other low-current devices on the robot.

The power for the motors first passes through a master 180A circuit breaker. From there, each motor is individually protected by a 40A, auto-resettable circuit breaker.

The power for the computer and the custom-made boards on the robot initially passes through a 60A manually-resettable circuit breaker, and then to a 6-position fuse block. The fuse block supplies power to four devices: the computer's power supply, the Power-over-Ethernet adapter, a Vicor 12V→12V DC-DC converter, and a Vicor 12V→5V DC-DC converter; each of these devices is protected by an appropriately sized fuse. The outputs of the two DC-DC converters are connected to a custom circuit board, which provides additional fusing and distributes the power to a number of 4-pin (12V, 5V, GND, GND) and 2-pin (5V, GND) Molex connectors.

Instead of connecting all sensors directly to the battery as was done last year, this year they were powered off a fused power distribution board with polarized Molex connectors. A standardized Molex connector with 5V, 12V, and ground was used throughout the robot. The power distribution board was powered by 12V-12V and 12V-5V switching regulators, connected to the battery via circuit breakers.

5 Software Systems

In keeping with the open nature of our project the decision was made to use Open Source software throughout and release our own work under the GNU General Public License, version 3. We leveraged a wide variety of Open Source programs and libraries, visible in everything from RATT's operating system to the LaTeXsoftware used to typeset this document.

5.1 Operating System

The current stable release — "lenny" — of the Debian Linux distribution was chosen as RATT's operating system. The primary reason for this choice was Debian's strong reputation for stability. However, in order to improve hardware compatibility we upgraded the Linux kernel to the latest stable version.

5.2 Programming Languages

Python and C were chosen as the languages RATT's custom software is written in. This decision was largely based on their mutual compatibility and broad library support. Python is used for non-speed-critical portions of the code, such as serial and GPS communications, where the bottleneck is I/O. C is used for components that must run as fast as possible, such as image processing. Libraries used include OpenCV, GMP (the GNU Multiple Precision Arithmetic Library), and the GTK+ widget toolkit. We also make use of the typical UNIX utilities as well as `gpsd`, a GPS device daemon; `AVLD`, a video loopback device, and `mencoder` to read the camera's RTSP stream.

5.3 Architecture

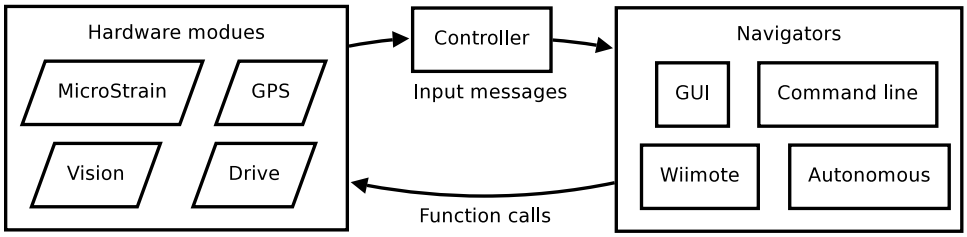

Figure 5: Software architecture diagram

The RATT software platform is build around a Controller which loads a variety of additional hardware and navigation modules required to control the robot. The Controller also serves as the primary means of routing messages between different modules.

Hardware modules
: Hardware modules serves as interfaces to the various hardware components. Currently hardware modules exist for the MicroStrain, GPS, motor controllers, and camera.

Navigators
: Navigators provide a means of controlling and monitoring the RATT. Current navigators include a graphical interface, command line interface, and Wiimote interface. In addition to these human controlled navigators the autonomous code is implemented as a navigator. The RATT can switch between different navigators on the fly; the one in control of the robot's movement is referred to as the *active* navigator.

Messaging and communication
: To facilitate communication between devices the Controller maintains a queue of input messages. Hardware modules generally send messages when new data becomes available for processing. One or more of the navigators can than receive and process the message. Once processed, the active navigator may choose to perform addition steps such as setting the motor speeds.

5.4 Image processing

RATT 2 relies on computer vision for both obstacle avoidance and lane following. We employ a multi-step solution that relies on machine learning, image recognition, and computer vision techniques. An overview of this processes is shown in figure 6.

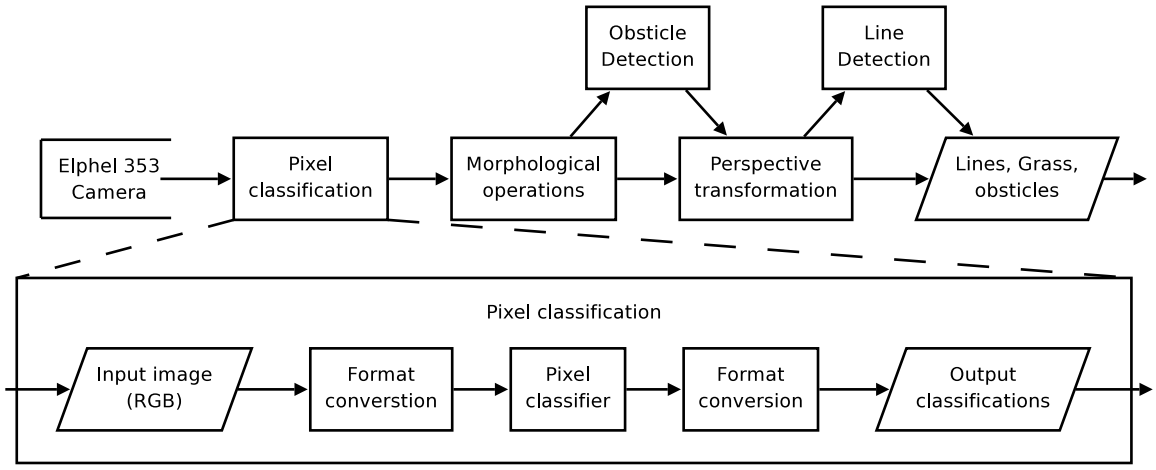

Figure 6: Diagram of image processing algorithms

Image acquisition
Images are obtained from the Elphel 353 camera using the Real Time Streaming Protocol and routed though a Video4Linux device. Our software then obtains the image using the OpenCV V4L image acquisition API.

Pixel classification
One of several pixel classification algorithms is then run on the transformed image. The output of this operation is a matrix of object types identifying each pixel as being part of an obstacle, a line, or the grass. Several of our pixel classification algorithms are detailed later.

Morphological operations
Testing has shown that the raw output of the pixel classifiers is fairly noisy. We post process these output images with a variety of morphological operations including erosions and dilations.

Obstacle detection
After classifying areas of the image it is important for our software to detect the base of any obstacles encountered. We currently do not employ stereoscopic vision algorithms or range finders. Instead, we rely on detecting the base of each obstacle being an accurate representation of the obstacles location on the field. While this works well for solid obstacles, "floating" obstacles such as the cross beams on the barricades are more challenging. To help mitigate this we have position our camera at a high elevation to get as much of a "top down" view of the field as possible.

Perspective transformations
The first step in processing the images is to perform a perspective transformation. This both scales the images to an appropriate resolution and and eliminates the perspective distortion caused by the camera being positioned at an angle. The MicroStrain is used to generate the transformation matrix based on the current orientation of RATT.

Line detection
While we can rely on our pixel classifier to produce accurate output for solid lines, the IGVC rules state that dashed lines may also be present as part of the course. To the gaps in these lines we calculate linear regressions on the existing lines and use their positions and slopes to fill in missing areas.

5.5 Pixel classifiers

The core of our image recognition is based around a set of pixel classifiers. All of these classifiers are written in C using the OpenCV image processing and machine learning libraries.

The simplest of these is a hard coded classifier that thresholds the input image using predefined ranges. This provides very fast computations but has the drawback of needing to be hand-tuned when the environmental setting of the RATT changes. In addition, a single ranges is used to classify grass and lines which can result in suboptimal performance.

More advanced classifier classifiers are implement using machine learning algorithms. Currently we are using decision trees for as our main classification algorithm but OpenCV also supports Bayesian classifier, K nearest neighbors, Support Vector Machines, and Neural Networks which could easily be used in place of decision trees. OpenCV also support boosting so a combination of simple classifiers could be employed as well. Our software takes the following steps when using machine learning classifiers.

1. Initialization: This chooses the underlying algorithm and sets up the classifier.
2. Adding data: Training data is provided by the users of the software. This data primarily comes from a set of hand classified images.
3. Training: Once all the training data has been loaded the classifier is trained to recognize patterns in the input that correspond to output classes. Once trained the classifier is ready to be used.
4. Prediction: Images from the camera are converted to the appropriate input space and sent to the classifier which then predicts output values for each pixel in the input images. These predictions are then converted back into a matrix of outputs with the same dimensions as the input image.

5.6 Autonomous Navigation

There are three primary functions of RATT's autonomous navigator. The first it to perform general obstacle avoidance while attempting to move to a specified location. The reaming to functions provide the goal location that RATT is to navigate towards. These two functions correspond to the autonomous and navigational challenges that make up part of the IGVC.

Obstacle Avoidance

RATT's autonomous code contains a map class which represents the course that is currently being navigated. This maps starts out blank and sectors of the map can be marked as bad sectors if they correspond to obstacles or boundary lines. RATT uses a combination of its computer vision system and its currently location as given by the GPS to calculate what sectors of the map should be marked as bad sectors.

In addition to storing bad sectors, the map also remembers previously visited areas so that RATT will not get stuck in repeating the same sequence of actions when attempting to avoid obstacles.

Obstacle avoidance itself is perfumed by determining which sectors between RATT and the goal location are obstacle free and then attempting to move though those sectors. When obstacles are encountered those sectors are marked as bad and RATT attempts to plot a new course.

Lane Following

Lane following is accomplished in the same way that obstacle avoidance is accomplished. The boundary lines on either side of the lane are marked as bad sectors and RATT attempts to stay away from them. When performing lane following the goal location is not not defined so RATT navigates by attempting to "flee" from the sectors that have been marked as already visited. In addition, when lane following, an attempt is make to move in a somewhat linear direction while staying as far away from bad sectors as possible.

Waypoint Navigation

When attempting the navigation challenge, RATT again uses the map to avoid obstacles, in this case the fence. The order in which the obstacles are to be navigated to is determined at the start of the round. When performing the navigation challenge RATT only uses visited information when a direct path path to the target waypoint is not available.

6 System Integration

During the Fall and Winter quarters the Hardware, Electronics, and Software teams worked independently in order to design and build their respective components. Each team scheduled meetings independently and worked at their own pace with an administrative meeting held once per week in order to determine the status of the various portions of the vehicle.

By the start of the Spring quarter the vehicle was starting to come together and we shifted the meeting schedule to better facilitate integration of the components. In addition to the weekly status meeting regular weekend "work days" were also held so that the teams could work together on tasks that did not fit nicely into one specific topic area.

6.1 Electronics Assembly

The major integration task was to mount the various electrical devices onto the frame constructed by the hardware team. This required interaction with the majority of the team. For example, the hardware team actually did the construction and mounting but with the position of devices such as the accelerometer and camera were determined by the software team. Finally, the electronics team was responsible for determining how the different components would be wired together.

6.2 Communication Protocols

By Spring quarter the electronics team was nearing completion on the motor controllers and specifications were needed so that they could be accessed from the software. For simplicity this was done with a simple Tag-Length-Value protocol over RS-232.

6.3 Integration Testing

Integration testing was performed incrementally as the various components were assembled. Our test platform, RATT 1, proved useful as well, as it meant that the integration testing of components like the software and motor controllers could be conducted before the RATT 2 frame was available.

7 Vehicle Purchases Summary

Component	Regular Cost	Total cost for team
Mechanical		
Motors	$28 × 6	$168
Gearboxes	$123× 6	$739
Wheels	$5 × 6	$30
Frame	$100	$0
Bearings	$14 × 12	$168
Pillow blocks	$5 × 12	$60
Fasteners	$200	$200
Computer box acrylic	$14 × 10	$140
80/20	$50	$50
Mast assembly	$50	$50
Encoder assembly	$19 × 2	$38
Inner hub	$3 × 6	$18
Outer hub	$3 × 6	$18
Battery/Payload box	$10	$0
Electronics		
Elphel 353 camera	$1,000	$0
Garmin 16x GPS	$95	$95
Microstrain 3DM-G	$795	$0
Computer	$315	$315
Batteries	$119 × 2	$200
Motor Controllers	$150 × 6	$170
Motor Controller Controllers	$35 × 2	$70
Kill Switch	$80	$80
Power Distribution	$200	$0
Vicor VI-J00	$104	$0
Vicor VI-J01	$104	$0
Encoders	$60 × 2	$120
Software		
Everything	$0	$0
Total	$5,809	$2,729