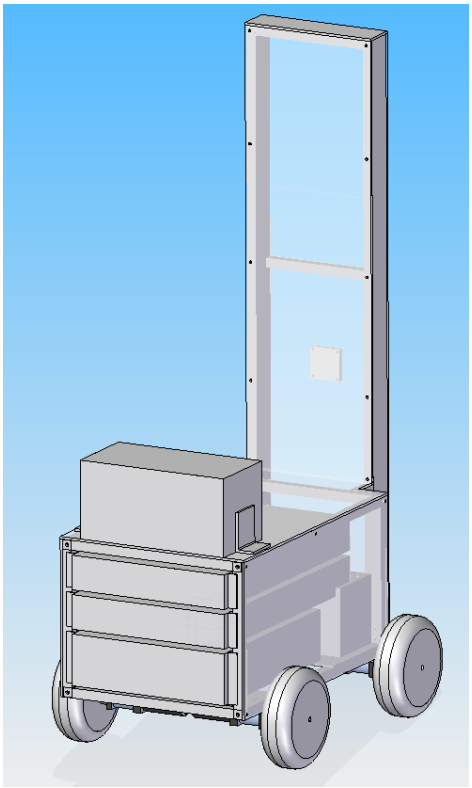
Moxom's Master

Nick Wahl, Trenton Tabor, Mark Jacobson, Thomas Most,
David Baty, Cameron Twarek, Jon Klein
{wahlnj,taborts,jacobm,mosttw,batydm,twarekcb,kleinjt}@rose-hulman.edu



Rose-Hulman Robotics Team, CM 5000

Rose-Hulman Institute of Technology

5500 Wabash Avenues

Terre Haute, IN 47803-3999

May 14, 2010

This report and the described vehicle were designed and constructed by the Rose-Hulman Robotics Team and the work done during the 2009-2010 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 and graduate 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 Moxom's Master (named after the short story by Ambrose Bierce). It reflects our growing experience with the IGVC, having attended the 2008 and 2009 competitions unsuccessfully.

2. Design Process

2.1 Team Organization

The president oversees the functions of the team and is responsible for setting goals and deadlines for the project, seeking sponsors, and other administrative tasks. Three sub-teams focus on the mechanical, electrical, and software components of the project. Each sub-team holds meetings each week, and the team holds a weekly administrative meeting to discuss the progress each sub-team has made as well as any administrative business. The team structure is shown below in Figure 1.

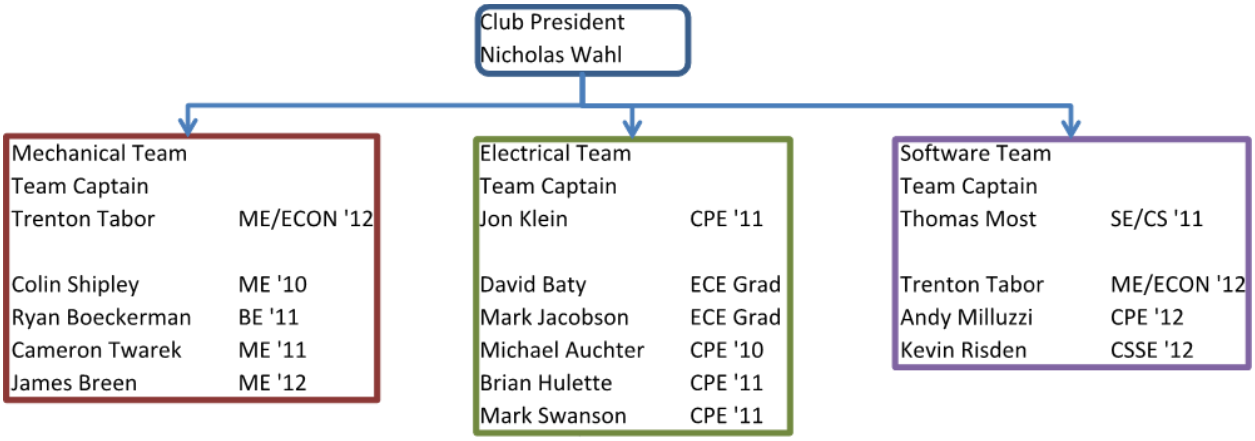


Figure 1: Team Organization

2.2 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 consists 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.

2.3 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 serviceable 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 accommodate the
necessary controller ICs.

2.4 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 previously, and our motor controllers, which have proven superior to the custom units that prevented us from competing in the past. In our software we try particularly hard to use existing solutions, detailed below.

3. Hardware Platform

3.1 Overview and Previous Designs

The mechanical design of Moxom's Master reflects the experience of the team gained from the previous two years of competition. It incorporates many of the elements of the previous designs that have worked well, as well as new features to correct issues with previous designs. The motors and gearboxes are easily interfaced with our control and power systems, and are also easily mounted in a variety of positions. In RATT II, optical encoders were connected to each motor via sprockets and chain and mounted in a protective housing. This system worked well, so these components were also migrated to Moxom's Master.

On the other hand, the mechanical and electrical complexity of a six-wheel drive proved to outweigh the benefits of increased power and control. In addition, the lack of a convenient and accessible mounting area for electronics and lack of an interface to control the onboard computer proved to be problematic. The custom-welded aluminum frame was time consuming to construct, and made it difficult to add or modify components throughout the year. All of these experiences and others were taken into account when developing Moxom's Master.

3.2 Design Considerations

Two primary designs were proposed: a two wheel drive robot with a pivoting caster in the rear, as well as a four wheel drive robot with two separate drive train assemblies. To decide between the two, decision criteria were formulated. With these, a decision matrix was constructed as shown in Table 1, scoring the two designs and rating them against one another. Of the criteria, stability, reliability, and maintainability were considered the most important, as shown by their higher weighting. Cost and simplicity of each of the three subgroups were also evaluated, but as they do not directly impact performance, were valued a lesser weight.

Table 1: Decision Matrix

Criteria	Weight	2 Wheel Design		4 Wheel Design	
Cost	2	7	14	5	10
Software Simplicity	1	3	3	2	2
Electrical Simplicity	2	5	10	3	6
Mechanical Simplicity	1	6	6	5	5
Stability	3	4	12	7	21
Reliability	3	7	21	5	15
Maintainability	3	4	12	3	9
Total			**78**		**68**

Advantages of the two wheel drive model included easier construction of the suspension, as only two springs would be necessary, as well as well-defined turning characteristics. Further, it was thought to be simpler overall, and to allow for easier configuration of software and electronics. The four wheel drive model was similar to the frame of the previous year's robot. Also, it would be able to supply more power, having two additional motors. Weight distribution would be improved, and frame construction would be simpler.

While the two designs tallied identical totals for the three performance-based criteria, the two wheel drive outscored the four wheel proposal in cost and the simplicity categories. For this reason, it was decided to build a two wheel drive robot. This decision was made with the plan that if for any reason this solution did not prove effective, a switch to a four wheel design could be made. These changes would be made easy by the fact that they would require no alterations to the frame of the robot, and consist only of adding two additional wheel and motors, parts already owned by the team.

After testing the robot with the two wheel drive configuration, two problems became evident. First, with much of its weight on the caster, the robot tended to swing around out of control when making a quick turn. In addition, the robot had difficultly starting to move on rough terrain and driving up an incline, indicating it was underpowered. The team made the decision to replace the rear caster with another motor assembly identical to the one already in use. This conversion has proven to solve the issues with the caster configuration.

3.3 Chassis

To facilitate easy mounting and organization of the electronics as well as easy access to the components, the team used three standard rack mount cases. This decision drove the basic design of the frame, a box which would accommodate these three cases. A clear view of the course ahead requires the camera, our main sensor for the course, to be located as high as possible. In addition, an easily accessible interface for the computer was needed. A wide, hollow mast at the rear of the robot proved a robust solution to both of these concerns, providing a mounting point for the camera, a monitor, and a keyboard. Since a smaller robot leaves more clearance for navigation of the course, the team designed the chassis to be as close to the minimum length of 36 inches and width of 24 inches as possible. Finally, the team sought a material that would be easy to assemble and modify, yet still be robust throughout operation. Many members had experience using 80/20, and the team already owned a large amount of 80/20, so this was a logical choice for the chassis.

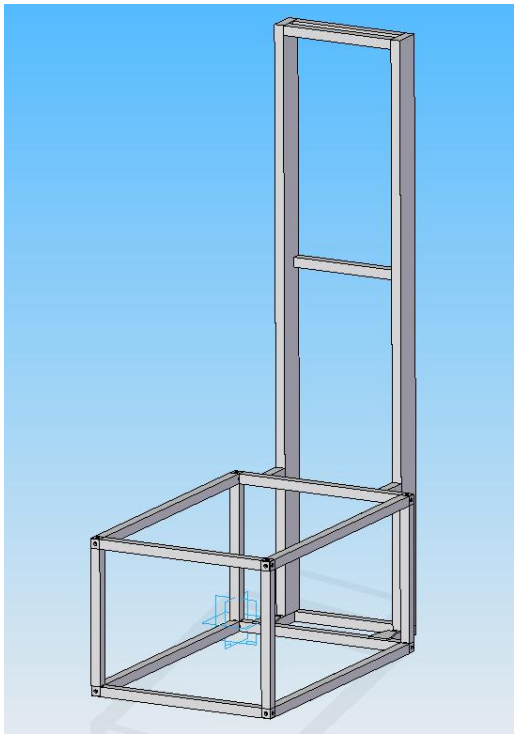


Figure 2: Frame design

3.4 Drive Train

While the team used the same motor and gearbox combination as the previous year, significant upgrades to the drive train were made. To prevent damage to the fragile electronic components, a suspension was designed. The motors and gearboxes were attached to a steel plate, from which the main frame of the robot was suspended by two springs. This provides a significantly smoother ride, damping out most shocks associated with traversing a grass course.

Due to the team's experiences the past two years, the current design of the drive train was known to be capable of propelling the robot at a velocity in excess of five miles per hour. Therefore, a hardware motor control system combined with information from optical encoders to limit the speed to no greater than five miles per hour, per contest regulations. Additionally, since the motors are not required to run at full power output in order to achieve this speed, it is known that unused power is available in order for the robot to climb an incline.

3.5 Summary

Building off previous designs, Moxom's Master better addresses the needs of the team and the requirements of the competition. The modular 80/20 frame allows for easy modifications to the design, as well as easy mounting of components to the frame. By designing the frame around the three rack mount electronics cases, the team allowed for easy access to and mounting of electronics. The suspension reduces the effect of vibrations on the computer and electronics. To allow the team to make changes to the software on the robot, a monitor and keyboard were integrated into a mast, which also served as a mounting point for the camera and other sensors. The completed design is shown below in Figure 3. While the design resolved many of the issues the team has encountered, it has presented its own set of challenges, and there is certainly room for improvement with subsequent designs.

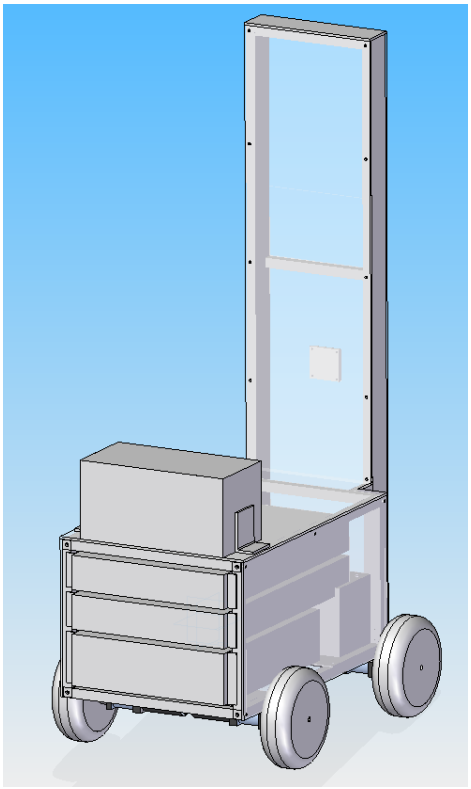


Figure 3: Completed Robot Design

4. Electrical Systems

4.1 Computers

As image processing is very CPU and memory intensive, the computer on Moxom's Master 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.

The operating systems and control software are installed on a 32GB solid state drive. Solid-state storage media was chosen for two purposes: to reduce the power consumption and
eliminate moving parts. The computer and solid-state drive are powered using an off-the-shelf 160W DC-DC converter.

The computer hardware has changed from what was used in the 2009 competition. The solid state drive replaced an unreliable 4GB compact flash drive. The case for the computer is also different. It is mounted in a 3U rack mount server case, like much of Moxom Master'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 the rack mount cases.

4.2 Motor Controller

For our 2010 entry, we decided to upgrade our motor controllers. The previous Victor 883s has a variety of issues, including the requirement to be controlled by 50Hz servo PWMs. This required a separate board to create these signals and interface with the CAN bus. The team is now using Texas Instruments Black Jaguar motor controllers (MDL-BDC24).

The Jaguar motor controllers allow for direct control over the CAN bus, as well as acting as CAN to RS232 bridge. They also can take direct inputs from our optical encoders and perform PID control with limited interaction from the computer. All of these settings including current motor voltage, current and PID parameters are accessible over the CAN bus. The new motor controllers also have the option to require a heartbeat signal every 100ms to continually power the motors. This allows us to create a very simple fail safe killswitch that sends the heartbeat to the motor controllers as long as the killswitch is in contact with the robot.

4.3 Encoders

The team has continued to use Grayhill 63R, 256PPR optical encoders. Two more have been added this year for a total of four, which allows us to control each motor much more accurately. Cables connected to the encoders were also upgraded this year after constant failures with them last year.

4.4 Wireless Emergency Stop

The emergency stop was designed to fulfill contest requirements of a remote kill switch. The kill switch is comprised of two devices: a handheld remote transmitter, and a complementary receiver on the robot. Each of these devices is comprised of an Atmel microcontroller paired with an off-the-shelf wireless Zigbee transceiver. The remote transmitter periodically sends a status packet to the robot, indicating whether the robot should be enabled or disabled. The receiving board on the robot sends a heartbeat out on the CAN bus if and only if the following conditions are all met: the physical kill switch is enabled, a valid status packet from the remote transmitter was received, and the value of the packet indicates that the robot should be enabled. Since the motor controllers require a 100 ms heartbeat to stay active, if any of these conditions are not met, the robot will be disabled.

4.5 Inertial Measurement Unit

The IMU on Moxom's Master 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 device is primarily used as a source of heading and rotational velocity information.

4.6 Camera

The camera used for line-detection and obstacle avoidance is a 5-megapixel 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, and have designed our image processing algorithm to be portable, at least in part, to the FPGA.

4.7 Power Distribution

The power system was completely redesigned this year. After failure of one battery, the team decided to upgrade them, as well as put them in parallel instead of using separate batteries for the motors and electronics. The team uses two 12v 60AH Power-Sonic sealed lead acid batteries in parallel to power the entire system. This parallel setup increases overall runtime and should result in significantly longer battery life. The likelihood of improper connections during servicing also significantly decreased due to the use of different size polarized connectors wherever possible.

This year, the power system was divided into two 2U rack mount cases to ease maintenance as well as fit several large new components. One case houses the power tethering circuitry, including the AC-DC power supply, battery charger, main battery breaker, and main-cutoff switch. Figure 4 shows the connections internal to the case, as well as the external hookups. The tether case takes in AC wall power and the 12V DC parallel battery connection and outputs separated 12V lines for the motors and electronics. The tie point for connecting earth, motor, and electronics ground is also in this case.

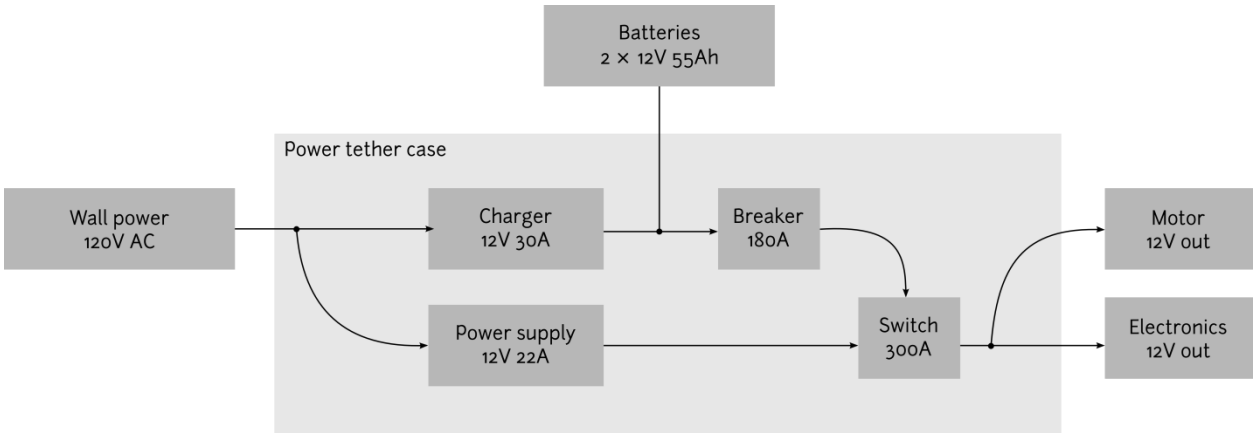


Figure 4: Power Tether Case Diagram

Inside the power tether case, AC power is routed to a 12V, 30A battery charger and a 13.5V, 22A power supply. The battery charger is connected directly to the battery inside the case and automatically charges as soon as AC power is applied. This hard-wired automatic charger will ensure that batteries are not over- or under-charged due to improper settings, a problem that has come up often with cheap external chargers. The 13.5V supply is used to power the on-board electronics while the batteries are charging and is enabled by switching the master switch to the secondary position. This ease of swap-over simplifies software testing and is designed to enable automated switchover with the addition of a single circuit board next year.

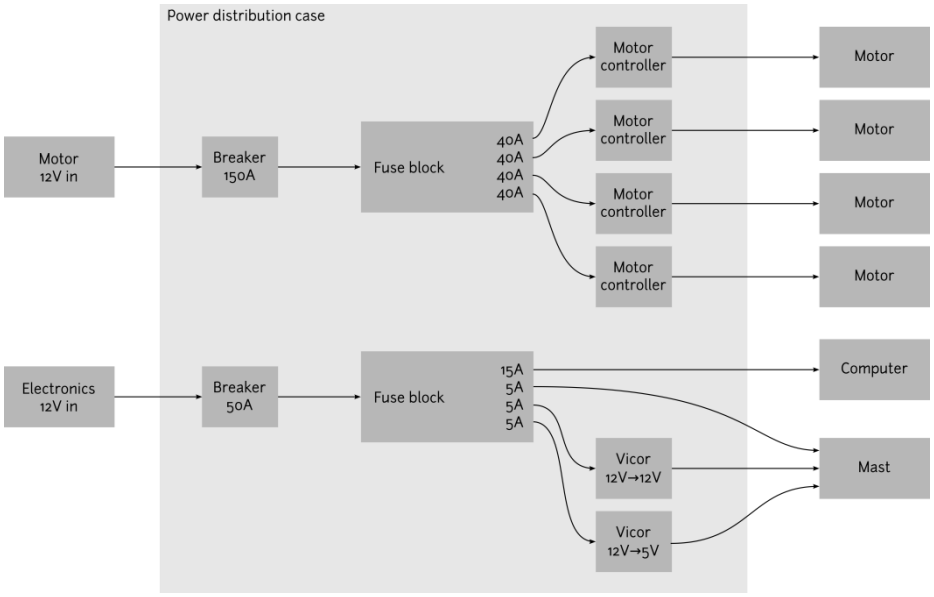


Figure 5: Power Distribution Case Diagram

The power distribution case takes in separate raw 12V DC connections for the motors and electronics in order to minimize noise. The motor power line is passed through a 150A manually-resettable breaker and into a quad fuse block with a 40A auto-resettable fuse for each motor controller. The motor controllers individually regulate the power passed to each motor and their outputs are passed directly out of the case to the motors themselves. The electronics input is passed through a 50A manually-resettable breaker into a fuse block that then supplies two Vicor DC-DC modules (12v to 12v and 12v to 5v), the computer, and the mast with raw battery power. The Vicor modules' regulated outputs are also passed to electronics mounted on the mast. Figure 5 is a wiring diagram for the power distribution case.

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. Rewardingly, some of our code (for capturing images from the Elphel camera) has already been of use to others. A wide variety of Open Source programs and libraries were leveraged in the development Moxom's Master, ranging from the robot's operating system to software used to interface with sensors.

5.1 Operating System

Moxom's Master is running the desktop version of Ubuntu 9.04. The switch from Debian to Ubuntu was made to gain easy access to more recent system software and drivers. At the 2009 IGVC disk corruption necessitated a system wipe and reinstall, but difficulty with custom-compiled packages (most notably mplayer, used to fetch images from the camera) meant

that the team was unable to recover from this failure. Ubuntu includes package versions recent enough to use without customization, meaning that the system can easily be reconfigured in the case of failure, as it is largely stock.

5.2 Programming Languages

The robot's software is written in a combination of the Python and C programming languages, as was the case with its predecessors. Python also makes it easy to prototype new functionality, being a concise and dynamic language. C is used for performance-critical code—specifically, image processing—and allows circumvention of a key Python limitation. This limitation, the global interpreter lock, which serializes Python operations at the bytecode level, normally prevents true shared-memory parallelism (in exchange for ease of programming). A C-language extension module can release the GIL and continue processing non-Python data structures. Thus, the most CPU-intensive operations—image processing—can be done in parallel with other operations. This mixture of languages also matches those used in introductory programming courses at Rose-Hulman, making it easier for new team members to get to work on the codebase.

5.3 Architecture

Past experience revealed the previous publish/subscribe design to be a premature abstraction. It has been replaced with a more direct model of components interacting directly with their dependencies. The data flow is as follows:

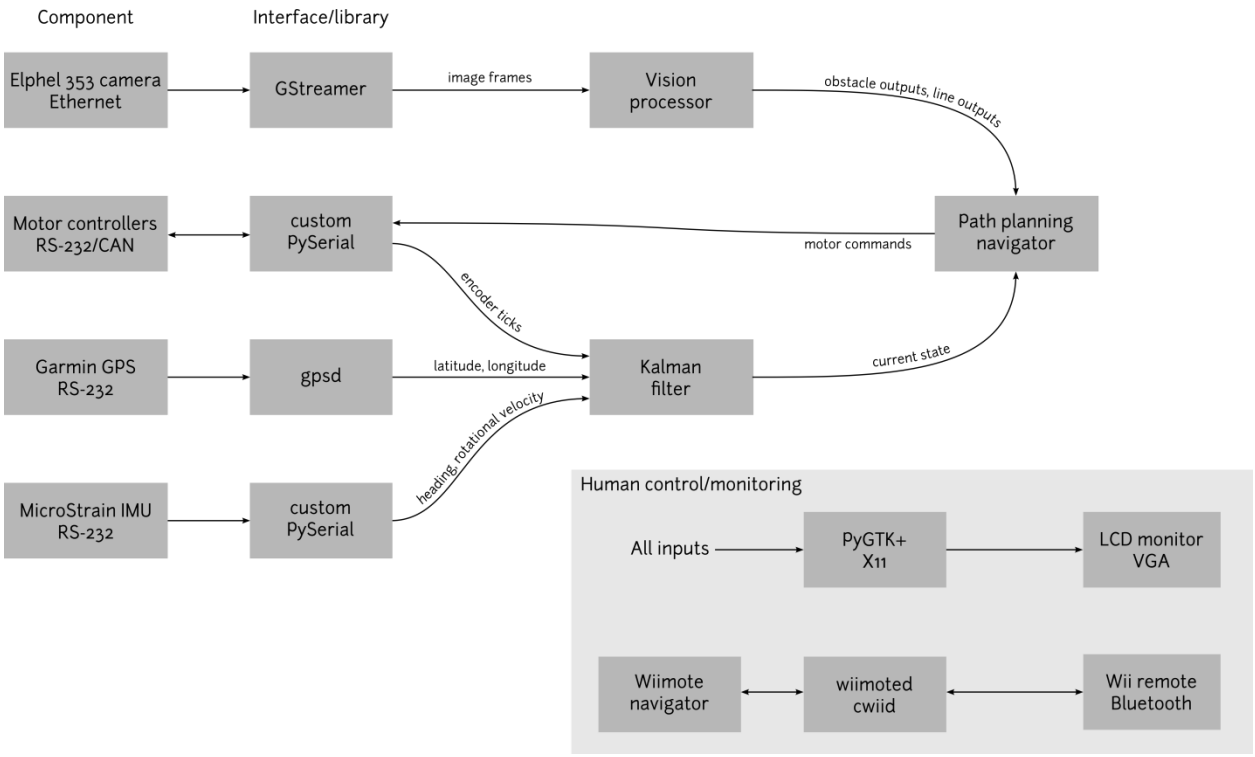


Figure 6: Software data flow diagram

Information flows from the sensors via various libraries or custom interface software into two processing components: the vision processor and Kalman filter. These output high-level data with is used for autonomous control in the path planning navigator. These systems are further detailed below.

5.4 Vision Processing

The robot's camera is its key navigational sensor, used for detection of both obstacles and lines. Thus, a large portion of development time was spent prototyping and implementing the image processing subsystem. Fundamentally, the algorithm is color based, differentiating obstacles and lines from the grass based on a sample of presumed grass at the bottom of the image. It produces a set of sparse points representing the 2D locations of the nearest obstacles, mapped to the ground plane via a projective homography. This algorithm has the advantage of being fast to execute and simple to implement.

Figure 7: Grass classification based on a "safe zone".

This algorithm is based on the work of Ulrich and Nourbakhsh (1), who explored a purely color-based model. Here there work is extended by including texture as a metric. This is made possible by the high resolution of the Elphel 353 camera. Note that most of the following calculations are done on downsized images (the scaling is elided for clarity), but that the Sobel filter is applied to the full five megapixel image.

Generate an HSV histogram of a "safe region" at the bottom of the image. This safe region is assumed to be grass.

Compare each pixel to this histogram, marking everything that does not fit the definition of grass marked as an obstacle (see Figure 7, above).

Apply a Sobel edge-detection filter to the value channel of the HSV image. Blur this, producing a grayscale indicator of the textural "roughness" of a pixel.

Mark each obstacle pixel where the roughness indicator exceeds a threshold value as a line. This is the line mask.

Find the bottom-most obstacle pixel in each image column. These are the obstacle points.

Projectively transform the obstacle points such that the ground plane is parallel to the view plane, also converting to physical units. These are the obstacle outputs.

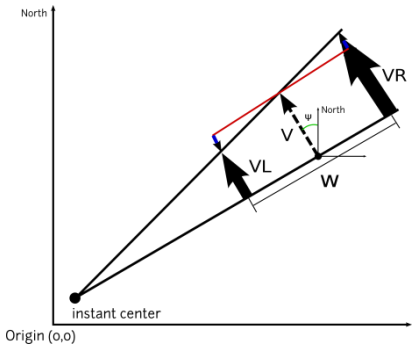
Apply a Hough transform to the line mask.

Projectively transform the resulting lines as in part VI. The resulting lines are the line outputs.

The key observation behind the method for distinguishing obstacles and lines is that the lines, painted on grass, are texturally "rough". In comparison, the plastic barrels and trash cans used as obstacles are smooth. Both of these classes differ significantly in color from the grass, so the three classes can be separated reasonably well. Note that it is more important to limit false positives on line pixels than it is to fully identify all lines, as small segments of false line are likely to cause problems when joining dashed lines later.

The reliance on a clear sample area of grass is both a plus and a minus here. Most importantly, it means that the algorithm is robust to changes in lighting. Testing has revealed that as the robot rotates the color and saturation of the captured images changes significantly, particularly in the early morning and evening. These lighting changes make exemplar-based methods that rely on color difficult to train, as large numbers of example images must be captured and classified to train the system. However, as the sample area is essentially an exemplar itself, if it is not in fact completely grass the entire output of the processing step is rendered invalid. Thus, a large buffer of space must be maintained in front of the robot to avoid this.

5.5 Robot Localization: Kalman Filter



We use a Kalman filter to account for noise from sensor measurements, and to take advantage of redundant measurements of the location or speed of the robot. Previously data from the sensors were taken to be ground truth, and information from redundant sensors was ignored. For example, while the wheel encoders, GPS, and accelerometer can be used to measure the velocity of the robot, only the wheel encoders were used. This section will first present on how our sensors relate to our model of our robot, then briefly describe Kalman filters, and how they will be used to improve the accuracy of the localization.

Robot as a Model

The robot has sensors for measuring the velocity, and absolute position and orientation. These sensors are not perfect. We model the imperfections of the sensors using noise. The Kalman filter assumes that noise is in a Gaussian distribution around the true value, so we need an estimate of the standard error of each of the sensors. In the figure the labeled vectors represent measurements and unknowns to estimate.

VL and VR are measured velocity from the wheel mounted encoders.

φ is the angle relative to north, measured by the digital compass and the GPS.

V is the absolute velocity of the robot, found by averaging VL and VR

Instant center is the intersection between the line perpendicular to the robot and a line connecting the VL and VR vectors; its distance to the robot is the radius of curvature

lat is the "latitude" of the robot, relative to its initial position

long is the "longitude" of the robot,

φ Is the heading of the robot in radians with respect to north, it ranges between 0 and 2π

V is the forward velocity of the robot.

ω is the rotational velocity of the robot.

These act as the states of our model:

$$x = \begin{Bmatrix} \text{lat} \\ \text{long} \\ \varphi \\ V \\ \omega \end{Bmatrix}$$

Brief Explanation of Kalman Technique

The idea of the Kalman filtering technique is that states are related to previous states times a transition, with noise added.

$$x_k = Ax_{k-1} + w_{k-1}$$

Where w_{k-1} is this noise term. Then measurements of states are related to the actual states added to a measurement noise.

$$z_k = Hx_k + v_k$$

Where v_k is the measurement noise.

Specific Matrices for Moxom's Master Localization

The transition matrix that relates the previous state to the current state is

$$A = \begin{pmatrix} 1 & 0 & \Delta t\cos(\varphi_{k-1}) & 0 & 0 \\ 0 & 1 & \Delta t\sin(\varphi_{k-1}) & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Each expression in the matrix relates some previous state value to current state values.

5.6 Autonomous Navigation: Path Planning

Moxom's Master generates a cost map from the vision data. Costs are based on ease of reach for every point; lines and obstacles are given infinite costs, points along the current line of travel are weighted least, and points requiring turning are weighted more. A goal point for each time step is generated by extending lines detected in each time step. With a discrete cost map the problem reduces to a 2-dimensional graph search. An A* search algorithm finds the best path, using a simple distance heuristic to estimate the cost of reaching the goal point from any point.

6. Integration and Cost Summary

6.1 Electronics Assembly

Assembly of the robot's electronics was a significant undertaking this year, as there were major changes to the robot's chassis and power systems. Electrical assembly tasks were focused on mounting and wiring the new power supplies and switching systems, as well as proper electrical conduction of the several sliding chassis to the frame for proper grounding. This required interaction with the mechanical team to properly accomplish, as minor modifications were required to ensure proper contact.

Another major integration hurdle was the addition of the large mast. The mast contains support electronics for the camera, LCD, keyboard, and killswitch, as well as mounting locations for them. Significant inter-team communication was necessary to correctly place all of the components on this mast correctly, coöperating on issues like camera angle and keyboard size and height.

6.2 Vehicle Purchases Summary

Component	List	Cost to team
Hardware		
Cases	$255	$85
Drivetrain	$400	$400
Acrylic Panels	$119	$119
Lubrication	$50	$50
80/20	$400	$200
Frame Hardware	$200	$200
Electronics		
Motor Controllers	$436	$436
Optical Encoders	$228	$228
Wire and connectors	$380	$330
Breakers, fuses, switches	$170	$170
Batteries	$330	$330
Tools	$56	$56
Miscellaneous	$100	$100
Battery Charger	$200	$200
Power Supply	$80	$80
MicroStrain 3DM-G IMU	$1,300	$0
Computer		
Elphel 373 Camera	$1000	$0
Total	$5074	$2984

7. Conclusion

Moxom's Master represents not only the culmination of work from 2009-2010 school year, but also the knowledge gained through participation in the IGVC the previous two years. Though unsuccessful, these two previous entries provided a vital foundation of experience upon which to build this year's robot. This paper details the design process which the team undertook in the development of all aspects of the robot: mechanical, electronic, and software. Through participation, the IGVC offers great educational benefit, including a great deal of applied engineering experience that cannot be gained in even the best of classrooms. For this reason, the Rose-Hulman Robotics Team awaits eagerly this year's competition, and competitions in years to come.

Bibliography

1. *Appearance-based obstacle detection with monocular color vision.* **Ulrich, I. and Nourbakhsh, I.** Cambridge, MA : MIT Press, 2000, Proceedings of the National Conference on Artificial Intelligence, pp. 866-871.