The Virginia Tech Autonomous Vehicle Team presents:

Johnny-5

Required Faculty Advisor Statement

I certify that the engineering design of the updated vehicle described in this report, Johnny-5, has been significant, and that each team member has earned six semester hours of senior design credit for their work on this project.

Charles F. Reinholtz
Department of Mechanical Engineering
Virginia Tech

1. INTRODUCTION

In the 2004 Intelligent Ground Vehicle Competition Johnny-5 won grand prize honors and in 2005 Johnny-5 placed third overall, with two other Virginia Tech vehicles taking top honors. Built and constructed over two years ago Johnny-5 has proven to be a reliable, durable, and functional vehicle for Virginia Tech. This year Virginia Tech is proud to enter a refined version of Johnny-5 in the 14th annual Intelligent Ground Vehicle Competition to compete in the Autonomous Challenge, Navigation Challenge, and Design Competitions.

Figure 1.1: The original cinema version of Johnny-5

New for this year's competition, Johnny-5 will be level-III JAUS interoperable, the highest level of interoperability within the JAUS standard. With such a reliable platform to build upon, Johnny-5 proved to be a prime candidate for JAUS implementation. The base vehicle, electrical system, and software algorithms have proven to be robust and reliable throughout years of rigorous testing, demonstrations, and competition. Because of this, software developers could focus solely on the challenging task of implementing level-III JAUS interoperability.

The vehicle's name originates from the popular 1986 cinema, *Short Circuit*, shown in Figure 1.1, which depicts a robot imbued with amazing humanistic decision making and control capabilities. Throughout the design process, the name Johnny-5 served as a constant reminder to design an autonomous system that could more closely mimic human intelligence and behavior.

2. DESIGN PROCESS

Each member of the design team has participated in previous IGVC competitions so they have first-hand knowledge of the requirements of competition and the track record of Johnny-5. Because the time-tested platform of Johnny-5 has proven to be reliable and rugged the design team focused mainly on implementing the highest level of JAUS interoperability on the vehicle. This implementation is by no means trivial, therefore the design team focused mainly on software improvements with only minor changes to the vehicle itself. With this in mind, the primary goals of providing a safe, reliable, durable, and competitive platform remained intact. To accomplish these goals, the team implemented a design strategy that held customer needs paramount, provided a clear path for project completion, and focused on innovations.

2.1. Team Organization

The design team is made up of graduate mechanical engineering students Brett Gombar, Andrew Bacha, and Ruel Faruque. Each graduate student participated in the design of Johnny-5 and assigned projects to a team of undergraduate volunteers. The team's structure is shown in Figure 2.1. Each graduate student also participated as advisors for Virginia Tech's newest vehicle Chimera and Gemini. A total of 300 student hours were spent on improvements to Johnny-5.

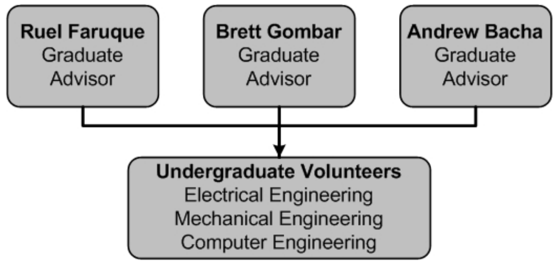

Figure 2.1: Johnny-5 team organization

2.2. *Target Customers*

The following primary customers were identified by the design team: (1) IGVC judges and sponsors who will evaluate vehicle performance, (2) the team faculty advisor who will evaluate overall vehicle design, (3) current and future vehicle users. Secondary customers include team sponsors and the autonomous vehicle community. Many of the primary customer needs were expressed in the 13th annual IGVC rules and the need to provide a reliable test platform for ongoing unmanned systems research at Virginia Tech.

2.3. *Design Planning Process*

A methodical design process is essential for the successful development of complex systems such as Johnny-5. The team used the Kano design method described in *Attractive Quality and Must-Be Quality Method* (Kano, Seraku, Takahashi and Tsuji, ASQC Quality Press, 1996) to guide the design process.

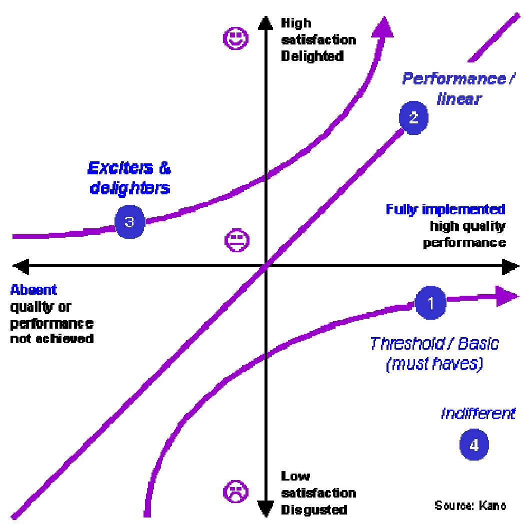

Figure 2.2: Kano design methodology

Figure 2.2 illustrates this simple common sense approach to the design process. For example, for a customer to be fully satisfied a product must first meet the basic "must have" needs. On Johnny-5 these needs include having a sensor suite capable of providing the necessary perception providing a reliable system for testing and evaluation of software, meeting all safety requirements, and complying with the 5 mph maximum speed limit. The Kano model predicts that customer satisfaction will increase linearly with improvements in performance parameters such as the maximum reliable navigation speed of the vehicle and the continuous run time for testing. Finally, the Kano model suggests that customer satisfaction is strongly enhanced by unexpected features that are not found in competing products, Kano refers to these features as delighters.

Kano's method allowed the design team to focus on the improvements necessary for increased customer satisfaction while continuing where the previous design team left off. We believe that the use of the Kano model provided a simple and efficient approach to the redesign of Johnny-5.

2.4. Establishing Target Specifications

Target specifications were established by determining vehicle performance requirements that fulfill customer needs. Performance requirements were determined by reviewing the IGVC rules and the performance of multiple IGVC vehicles. Based on this review, the design team determined that implementing level III JAUS interoperability while maintaining championship performance would be a worthy challenge.

3. Mechanical Design

The mechanical system of Johnny-5 has proven to be reliable and functional during competition and throughout extensive testing. Johnny-5 has served as a research platform for numerous projects at Virginia Tech. This extensive use and abuse has turned up no mechanical problems in the base chassis or drive train of Johnny-5. Therefore, the mechanical systems of Johnny-5 received very minor modifications.

3.1. Vehicle Chassis

Johnny-5's chassis, shown in Figure 3.1, has proven to be durable and functional after years of constant use and abuse. For competition, only the addition of supports to the mast assembly were made. The entire mast and laptop tray are supported by two 1 inch square tubes welded to the base frame. The addition of the supports eliminates the possibility of these welds failing due to fatigue. The final chassis is rock solid and measures 25 by 35 by 8 inches with a competition height of 70 inches. Two 16 inch rear drive wheels and a 10 inch front caster wheel provide Johnny-5 with a ground clearance of 3.75 inches. The chassis is constructed from welded 1" 6063 aluminum tubing chosen for its lightweight and nonferrous characteristics which reduce magnetic interference with the digital compass. A 1/16 inch 6063 aluminum plate covers the bottom of the frame and plastic panels cover the sides. Finally, an aluminum cover protects the onboard equipment from the elements.

Figure 3.1: Johnny-5 chassis

Placing the caster wheel in the front of the vehicle allowed for much greater control of where the vehicle's body would travel. In the past, caster wheels located in the rear of the vehicle would tend to swing out and collide with obstacles that have already been passed. The rear wheel drive design and weight distribution aids in traction as 60% of the weight is in the rear of the vehicle. Care was taken to mount heavy components such as the generator and batteries lower in the vehicle to lower the center of gravity, improving Johnny-5 maneuverability and stability.

3.2. *Vehicle Drive Train*

Johnny-5 is driven by two QuickSilver Control Silvermax 34HC-1 drive systems. Figure 3.2 shows an exploded view of the right drive train (a) and the assembled rear drive (b). The components of each drive system include a 16 inch composite drive wheel, a 10:1 NEMA 34 gear head, a 1/8 inch steel mounting plate, a Torrington PT Survivor bearing, and a custom drive shaft for each wheel.

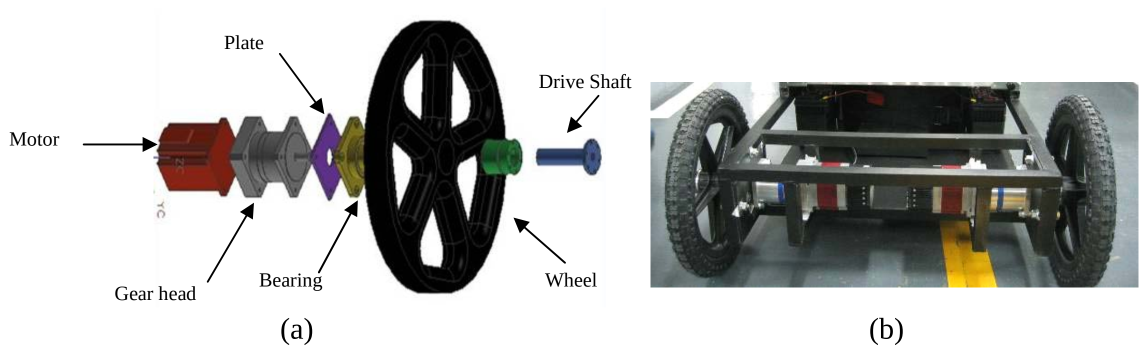

Figure 3.2: CAD model of right drive train (a) and installed drive train (b)

3.3. *Motion Control System*

The control system of Johnny-5 was simplified by using the integrated Quicksilver drive system. The motor interface is handled through a single RS-232 communications line. Each drive system uses an internal servo loop algorithm called Position, Velocity, Feedback/Feedforward, Integral, and Acceleration Feedback/Feedforward (PVIA). There are 7 gain parameters and 3 filter parameters that can be varied in this control algorithm. Testing revealed that Johnny-5 performed well with the default gain settings.

4. ELECTRICAL SYSTEM

Safety, reliability, COTS parts, and compactness are the principal goals of the electrical system. The electrical system provides communication between the computer, sensors, and motors as well as power to all on-board devices. Safety was addressed by creating a detailed electrical schematic, shown in Appendix A, using COTS automotive components, and implementing two emergency stop systems. Each wire is color coded according to voltage with red wires carrying 12 Volts, blue wires carrying 24 Volts, and black wires being ground wires.

The use of COTS automotive components was motivated by the need to purchase replacement parts at any local automotive repair shop. Additionally, automotive components can better withstand the dynamic conditions experienced on the vehicle. Finally, rugged mil-spec environmentally sealed connectors are used to connect communication and power lines to the electronics plate. The completed system is shown in Figure 4.1.

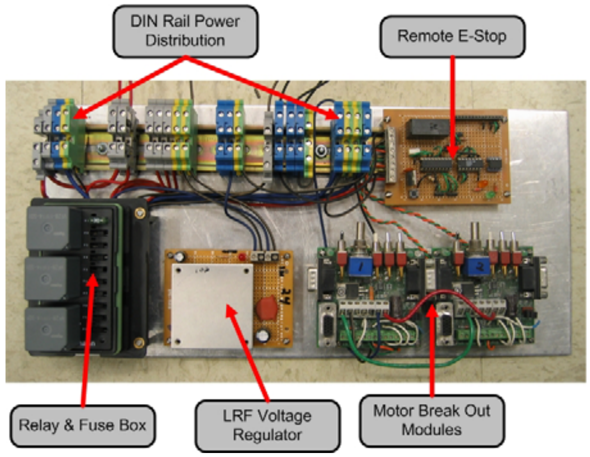

Figure 4.1: Johnny-5 power distribution panel

4.1. Johnny-5 Power System

Johnny-5's power system, shown in Figure 4.2, consists of two Hawker Odyssey PC535 dry cell batteries, a Soneil 24V 8 amp battery charger, and a Yamaha EF1000iS generator. The Odyssey PC535 is a sealed dry cell battery that recycles its internal gas during operation and charging. This increases safety while providing a long lasting power source. An insulative coating was applied to the battery terminals to prevent accidental shorting.

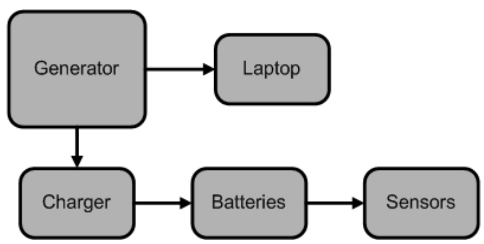

Figure 4.2: Block diagram of Johnny-5s' hybrid power system

The generator has a dry weight of 27 lbs and will produce 900 watts of power. As shown in Figure 4.2, the generator powers the laptop and charger, which in turn continuously charges the batteries. The batteries provide power to all the sensors and motors through an arrangement of switches, fuses, and voltage regulators. Care was taken to select sensors which could withstand a wide voltage range which eliminates the need for multiple voltage regulators. Johnny-5 only contains one voltage regulator to power the laser range finder due to its sensitive input voltage requirements.

4.2. Efficient Use of Power

The experience and recommendations of previous Virginia Tech IGVC team members contributed to the design and development of an effective and efficient vehicle power system. By incorporating a Yamaha EF1000iS generator, a Soneil 2416SRF battery charger, and two Odyssey PC535 batteries, Johnny-5 can sustain run times of up to 10 hours before refueling is necessary. These extended run times are achieved using the charge sustaining hybrid power system onboard Johnny-5. This means that the generator/charger combination can charge the onboard batteries faster than they can be discharged. Additionally, the Yamaha EF1000iS generator independently adjusts engine speed to match power demand, resulting in greater fuel efficiency and reduced noise.

5. SENSORS AND SYSTEM INTEGRATION

Electronic sensors and a laptop computer are used to gather course information, process the data, and decide the vehicle's path. Four sensors are used to obtain peripheral data. Table 5-1 briefly explains the primary function of each component and how it is used in the Autonomous or Navigation Challenges.

Table 5-1: Sensors used on Johnny-5

Sensor	Function	Characteristics
Unibrain Firewire Digital Camera 	Line Detection	Resolution: 640x480 94° Diagonal Field of View Update Rate: 15 FPS
SICK LMS-221 Laser Range Finder 	Obstacle Detection	Resolution: 1° 180° Field of View Update Rate: 15 Hz Accuracy: ± 5cm (range 3.2-65.6ft)
Novatel Propack LB+ DGPS 	Vehicle Localization	Accuracy: 15 cm 99% using Omnistar correction service Update Rate: 2 Hz
PNI TCM2-20 Digital Compass 	Vehicle Heading	Resolution: 0.1° Accuracy: ± 1° when tilted Update Rate: 15 Hz

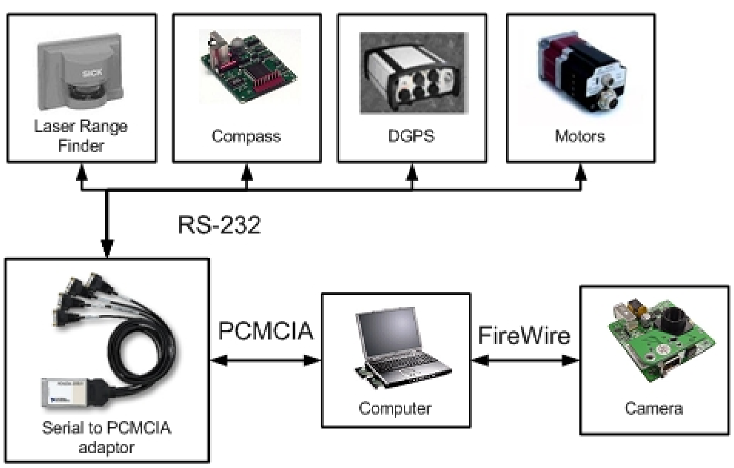

Figure 5.1: Sensor communication integration on Johnny-5

Figure 5.1 illustrates the communication protocol used for each sensor and its path to the onboard computer. Integral to providing the computer with the ability to individually address each sensor is the

PCMCIA to serial adaptor. This component is simply placed in the laptop PCMCIA slot and provides four RS-232 communications lines for each sensor. In the original design a RS-232 to USB converter was used to interface the serial devices to the computer. During testing, this converter was found to malfunction unexpectedly. Due to the difficulty in adequately resolving the issue, the design team decided to replace the converter with National Instrument's PCMCIA-232/4 serial to PCMCIA 4 port adaptor. Testing the current system has shown the serial to PCMCIA adapter to be reliable.

5.1. Sensor Communications and Data Integration

As described above, each sensor has an independent line of communications to the computer. However, care must be taken to analyze the data coming from each sensor correctly. For each sensor a custom driver was written to output the most recent complete data frame. This prevents a partial message from corrupting navigation algorithms and aids in error checking. Once valid data is obtained, it is combined into a common coordinate frame centered on the vehicle and shown in Figure 5.2. Once in a common coordinate frame, the data can then used to navigate the vehicle. Aside from converting sensor data to a common coordinate frame, there is no additional signal processing that must be done. The laser range finder, differential GPS, digital compass, and motors on Johnny-5 have signal processing built into their hardware.

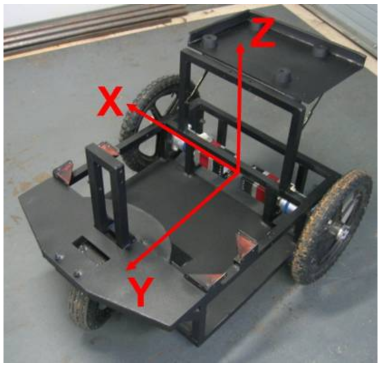

Figure 5.2: Johnny-5 coordinate frame

6. SOFTWARE

All software running on Johnny-5 was developed using National Instruments LabVIEW 8.0. Last year, the Autonomous Vehicle Team of Virginia Tech experienced great success with LabVIEW and has decided to standardize on this language for all programming needs. The use of LabVIEW simplifies coding and expedites system development. The graphical nature of this programming environment allows new team members to begin developing code with less formal training and experience.

6.1. IGVC Simulator

Software debugging and quantifying software performance are difficult tasks that have challenged developers of autonomous systems. To address these issues, Virginia Tech continues to make extensive use of a custom-developed simulator for software validation, benchmarking, and optimization. A key benefit of the simulator is the ease of migrating code to the actual vehicle. The AVT Simulator is a

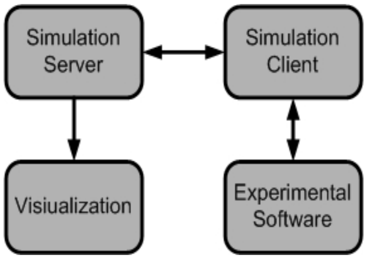

Figure 6.1: Simulation software structure

software library that allows the user to construct a virtual world with simulated lines, obstacles, waypoints, and vehicle dynamics mimicking previous IGVC challenges. Figure 6.1 illustrates the basic structure of the simulator software.

Also included in the simulator library is a vehicle creator, which allows the user to create an Ackerman or differential drive vehicle with sensors placed at user-defined locations on the vehicle. Dynamic properties of the vehicle can also be set.

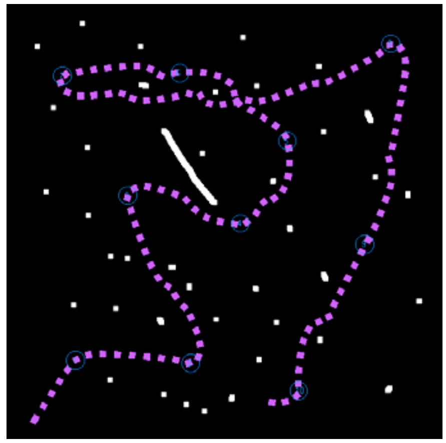

Figure 6.2: Simulated navigation challenge with **v**ehicle path (purple), waypoints (blue circles), obstacles (white)

Visualization, sensor simulation, vehicle motion, and recording options are all handled by the simulation server, providing virtual sensor outputs and receiving virtual motor commands via the simulation client. The prototype software, created by the software designer, interacts with the virtual sensor interface from the simulation client and allows the designer to test programs in an idealized virtual world. A global map of a simulated environment generated by the software is shown in Figure 6.2. The simulator eliminates downtime due to vehicle maintenance, poor weather conditions, and multiple software designers vying for vehicle testing time. Once the prototype software has been proven and debugged on the simulator it can then be transported to the vehicle for further testing.

Finally, a set of JAUS simulator components has been developed allowing the simulator to interact with the navigation software exactly as the vehicle software would. This enables the software to be tested in the same configuration, both in the simulator and on the vehicle.

6.2. Software Structure

To simplify software development, the programming structure shown in Figure 6.3 was implemented in both the Autonomous and Navigation Challenges. This structure is primarily reactive and does not rely on complex global map building to navigate. Rather, a series of behaviors is used to determine the vehicle path in response to sensor data. Sensor data is collected simultaneously through individual communication channels and processed by the navigation algorithm to determine a desired vehicle heading. Once a desired heading is computed, the navigation software plans a path for the vehicle around any detected obstacles. This subsumption architecture is implemented in both the Autonomous and Navigation challenges. Once the desired vehicle path has been determined, the motor control software executes the corresponding velocities for each wheel.

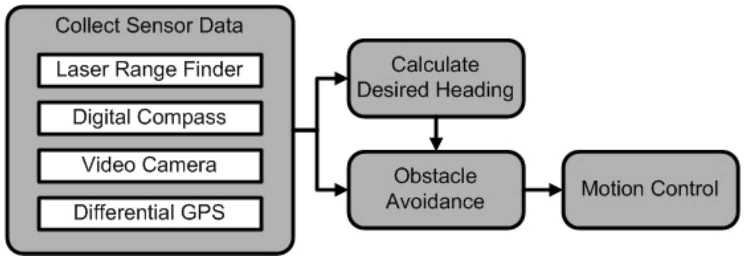

Figure 6.3: Software architecture for autonomous and navigation challenges

To facilitate code reuse and modularity, level-III JAUS interoperability is implemented in both the Autonomous and Navigation Challenges. This process was greatly accelerated by use of a LabVIEW-based JAUS development toolkit developed at Virginia Tech. An additional design process also developed at Virginia Tech was followed to convert the previous year's autonomous software to be JAUS-interoperable. This process will be discussed in Section 7. Due to the level-III interoperability, it will be trivial for future users of Johnny-5 to add software modules to interact with, to subsume, or to run in parallel with Johnny-5's current navigation software.

6.3. Autonomous Challenge Lane Following

The software programming to keep Johnny-5 within the course boundaries is a fairly simple lane-following algorithm. As shown in Figure 6.4, the lane following task can be broken down into subtasks consisting of detecting the lines, analyzing the lines, and setting the desired travel direction.

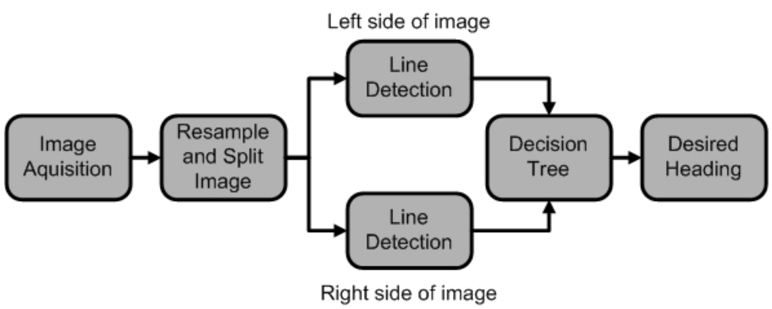

Figure 6.4: Autonomous challenge software diagram for calculating the desired heading

Once the image has been acquired it is resampled from 640x480 to 160x120 pixels, passed through a threshold operation, and divided into left and right halves. Respectively, these three steps are intended to reduce processing time, eliminate noise, and facilitate a structure for the line detection algorithm. Lines are detected using an algorithm known as the Hough Transform. Figure 6.5 shows the result of the Hough Transform used on an image of a line on grass after a threshold operation. Notice that the Hough Transform is not affected or skewed by the noise to left of the line in Figure 6.5. The result of the Hough transform is a score indicating how many points are on the line and an equation, giving both the location, and direction of the line. This information is then passed to a decision tree which determines the best direction for the vehicle to move towards.

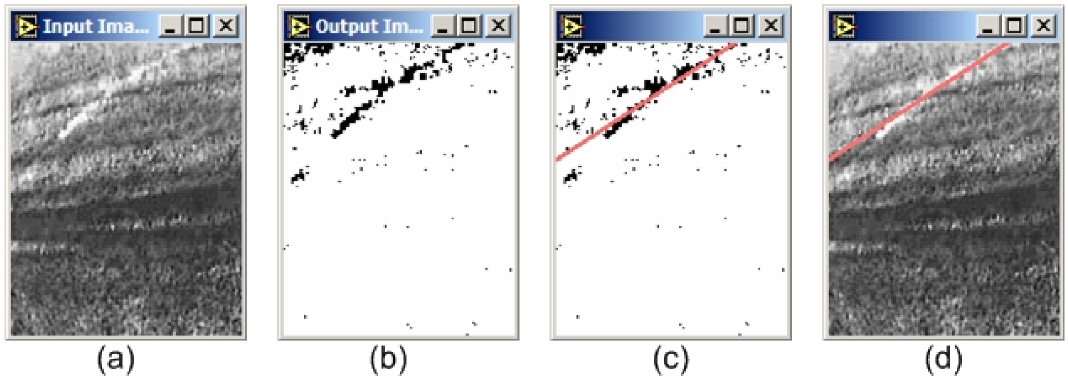

Figure 6.5: Original Image (a), After Threshold (b), Detected Line (c), and Original Image with Detected line (d)

The decision tree takes into account the line score and line orientation determined by the Hough transform. A low line score may indicate that the image identified noise as the line, while a high line score indicates a prominent line in the image. Orientation of the line will help determine the desired direction of the vehicle. If no line is detected, the software will decide if the line has become dashed or if the line has left the camera view depending on the last position of the line.

To set the desired direction of the vehicle, the detected lines are first corrected for perspective distortion. If the image contains both lines, the desired direction is set so the vehicle will head to the center of the lines. If only one line is present, the software assumes the lines are 8 feet apart and sets the direction based on the position of the known line. The desired heading, as well as the location of the lines is then passed to the obstacle avoidance software.

6.4. Autonomous Challenge Obstacle Detection and Avoidance

The obstacle avoidance process begins with mapping detected obstacles into 3x3 inch squares in an occupancy grid. During the Autonomous Challenge, the equations of lines generated by the Hough Transform are considered obstacles. Johnny-5 also considers distinct regions containing more than 80 white pixels as potholes. Since the camera can only detect lines 8 ft away from the vehicle, the laser range finder range is also limited to 8 ft. The lines and potholes detected by the camera are mapped into the same occupancy grid as the laser range finder data, resolving all the data into a common form.

Johnny-5's software then examines 36 potential arc shaped paths, beginning with the path closest to the direction determined by lane-following. An obstacle will lie in the vehicle path if the distance from the obstacle to the arc's center is between the turning radii of each wheel, as indicated in Figure 6.6. Each path is checked for obstacles, and the final path of the vehicle is chosen by combining the following factors for each path: distance to closest obstacle along path, deviation from desired heading, and the deviation from the last heading chosen. Once a suitable path selected, motor

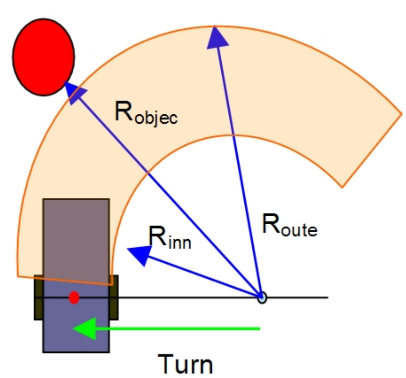

Figure 6.6: Distances used to calculate vehicle path to avoid obstacles

velocities are calculated and commanded to the primitive driver component, via JAUS Set Wrench Effort message.

6.5. Navigation Challenge Software

Johnny-5 placed second in the 2004 IGVC Navigation challenge, but suffered due to unreliable software. In 2005, the navigation software was redesigned, and Johnny-5 placed third, behind two other Virginia Tech vehicles. For the 2006 IGVC, this software has been made level-III JAUS-interoperable, using a waypoint driver shell tested by Virginia Tech at two JAUS Interoperability Experiments held by the JAUS Working Group.

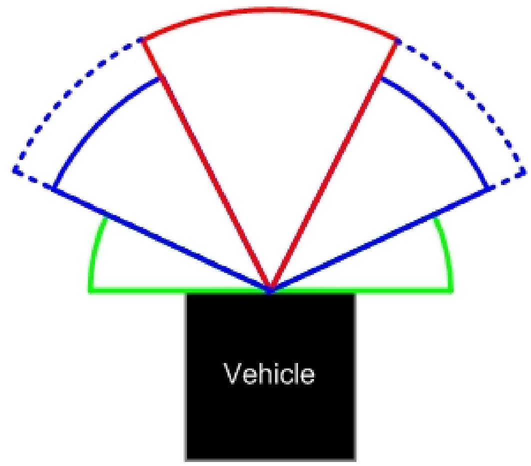

Figure 6.7: Obstacle avoidance regions for the navigation challenge. Data is from a single laser range finder scan.

At its core, Johnny-5's waypoint navigation employs a simple potential fields approach, embedded in a subsumption architecture. First, a desired direction is computed using the vehicle's current location and heading and the location of the next goal waypoint. Using this information, the vehicle will then travel directly towards the waypoint. If an obstacle is encountered while traveling to the waypoint the obstacle avoidance software then takes over motion control. Once the obstacle has been cleared, potential fields waypoint navigation is then resumed.

Where Johnny-5's original 2004 obstacle software relied on a local-map-based approach, the current software employs a simpler behavior-based avoidance strategy. The forward-mounted laser range finder is used to obtain a polar plot of the obstacles in front of the vehicle. Next, analyzing the predefined regions in front of the vehicle, depicted in Figure 6.7, the vehicle determines which direction to turn based on which region contains an obstacle and the direction to the goal waypoint. The first priority is to avoid obstacles in the center (red) region, followed by the middle (blue) and side (green) regions respectively. Testing in the simulator indicated a tendency for the vehicle to oscillate away from then back into an obstacle that was detected in the center region. To correct this, the two middle regions are grown out to the length of the center region and an obstacle that was avoided in the center region would then transition into a middle region and no longer turn back towards an obstacle. This is indicated by the dashed blue lines in Fig 7.7.

Testing with both the simulator and the real vehicle has proved this algorithm both robust and simple to implement. The emergent behavior from this algorithm is a tendency for the vehicle to round obstacles or follow along a wall of obstacles until a clear path to the goal is reached. We are confident that this reactive, behavior-based cone-avoidance algorithm will once again perform well in the 2006 IGVC.

6.6. Navigation challenge: GPS Loss and True North

For the GPS-loss portion of the navigation challenge, the team is expecting a short tunnel of some sorts. In the event of GPS loss, Johnny-5 will maintain its previous desired heading from potential fields, and rely primarily on obstacle avoidance to clear the tunnel walls until GPS is reacquired. If Johnny-5 senses that the GPS outage has extended too long, a Hough transform is calculated on the laser range finder data to determine the tunnel walls, and dead-reckoning used to exit the tunnel and reacquire GPS.

The True North portion of the competition is fairly straightforward. Since Johnny-5 is a differential drive robot capable of zero radius turns, it will simply stop at the last waypoint and execute a zero radius turn until aligned with true north. To address the problem of magnetic declination, the declination value will be determined using latitude and longitude coordinates of the last point and hard-coded into the digital compass. As a result, the data from the digital compass will be referenced to true north.

7. JAUS IMPLEMENTATION

Johnny-5 is level-III JAUS-interoperable, meaning that all communication between JAUS components on Johnny-5 takes place via JAUS messages. Theoretically, a JAUS-interoperable software module, for example a reactive obstacle avoidance module designed by Air Force Research Labs or University of Florida, could be installed on Johnny-5 and operate, with only a few changes to settings on Johnny-5's GUI. Furthermore, Johnny-5's navigation driver component can be implemented or tested on any other differential-drive JAUS-interoperable vehicle.

7.1. Learning JAUS

As one of the team members is involved with the Experimentation Task Group of the JAUS Working Group, Johnny'5's development team understood JAUS intimately before beginning the software modifications. Two new lessons arose however, in applying JAUS to an existing autonomous vehicle.

At its core, JAUS is a message set, not an architecture, despite the appearance of the word "architecture" in the acronym JAUS (Joint Architecture for Unmanned Systems). It does not specify the logical interactions of software modules in an unmanned system, only the language of those interactions. For this reason, conversion of an autonomous vehicle to JAUS-interoperability should not affect the logical interaction between software modules, only their physical interaction. JAUS can be understood as a language, and the conversion process as the installation of translators.

Experiments conducted by the JAUS Working Group have shown, however, that the JAUS language is not complete. While it has the words (messages), there is yet insufficient grammar (protocol) to enable smooth communications between separately developed unmanned systems. For example, one robot may expect a Set Waypoint message before a Set Travel speed message, and vice versa. The JAUS Working Group is presently addressing this issue.

7.2. *Integrating JAUS into Johnny-5*

In implementing JAUS on Johnny-5, a design process was closely followed, consisting of the following phases:

<u>Phase 1</u>: Define Interoperability Goals

- It should be possible for future developers to add JAUS components to Johnny-5 without modifications to the existing software. Since Johnny-5 has only one computer, this necessitates level-III compliance.
- When new JAUS components are added to Johnny-5, they should be automatically detected via the dynamic configuration messages.
- Johnny-5 should be interoperable with the IGVC Operator Control Unit (OCU).
- Johnny-5 should be interoperable with the Virginia Tech JAUS OCU.

<u>Phase 2</u>: Consider Current Software Structure

Previously, all of Johnny-5's software functions were contained within one software loop, inside one LabVIEW program, as shown in Figure 7.1. This monolithic structure, while efficient, does not allow for easy addition of new software modules without knowledge of the data structures and timing in the existing code. In essence, this single program became vehicle-specific, and could only be ported to other vehicles with the same hardware.

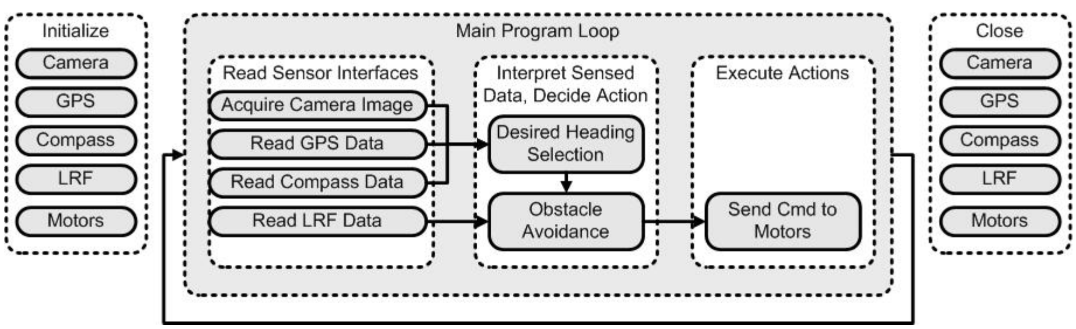

Figure 7.1: Previous software structure for Johnny-5. Single monolithic navigation program containing the vehicle-specific sensor/motor interfaces.

<u>Phase 3</u>: JAUS Component Definition

The primary goal in defining the components on Johnny-5 was to allow for code sharing with other autonomous vehicles at Virginia Tech. To accomplish this, the JAUS component boundaries were set between the vehicle-specific and non-vehicle specific components (with the exception of the camera interface, to be discussed later). In this way, a non-vehicle-specific navigation software module interacts with the vehicle using only JAUS messages, and thus can be ported to any JAUS-interoperable vehicle.

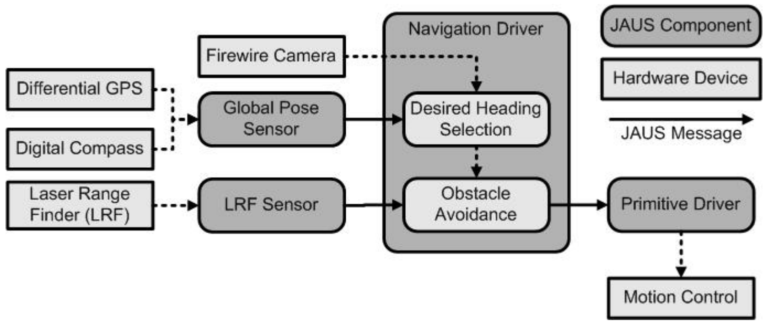

Figure 7.2: Level-III JAUS Implementation for Johnny-5

In addition, the monolithic structure of Johnny-5's previous software is eliminated. Since each JAUS component is a separate program, data sharing is simplified, as shown in Figure 7.2. For example, one could add a rollover prevention component to filter commands from the obstacle avoidance module to the primitive driver. It would be trivial for this software module to request attitude data from the global pose sensor – via JAUS messaging – with no modifications to the existing code.

Phase 4 and 5: Testing

Throughout the conversion, the "JAUS-ification" was tested out in the competition simulator, to verify that the performance characteristics were retained. In addition, the vehicle software was tested for JAUS interoperability against the Virginia Tech JAUS OCU, originally designed to verify the vehicle software used at the JAUS Experimentation Task Group interoperability exercises.

7.3. JAUS Challenge for IGVC

For the JAUS Challenge at IGVC, all messages from the IGVC OCU are received and handled by the Primitive Driver component. A simple state machine (Standby/Ready) is controlled by the JAUS messages 0004 Resume and 0003 Standby. In Ready, the navigation driver commands are executed, and in Standby zeros are commanded to the motors. Receipt of the Set Discrete Devices message activates a horn sound, likely from the laptop sound card. A speech generation routine announces any messages received from the IGVC OCU.

7.4. Challenges of Implementing JAUS

Largely due to the availability of the Virginia Tech JAUS Toolkit, very few challenges were encountered. The two most prominent issues were both related to messaging delay.

As described above, JAUS messages are used for all interactions between vehicle-specific and non-vehicle specific software modules – except for the camera interaction. The decision was made to retain the image acquisition process inside the Navigation Driver, to avoid the overhead of packing and unpacking camera images to and from the JAUS format. The associated delay would have been especially detrimental in the autonomous challenge, where the high-speed driving necessary to be competitive demands lower cycle times.

The second issue discovered relates to motor commands. When the navigation driver sends wrench effort messages to the primitive driver at too high a rate, the commands are queued in the primitive driver, leading to an accumulation of lag. One solution is to simply decrease the cycle time of the primitive driver, however the team chose to implement a command service connection, to ensure that only the most recent commands are executed by the primitive driver. This problem was not encountered with sensor information, as the navigation driver obtained sensor data via JAUS service connections, which inherently retain only the most recent data.

8. PREDICTED PERFORMANCE AND TESTING

The success of Johnny-5 is due in large part to extensive testing and years of use and abuse. Only with by using the system can unexpected flaws in the design be found and fixed. To date, Johnny-5 has undergone a number of upgrades to improve the base vehicle, software, and electrical system.

8.1. Vehicle Speed

The two Silvermax 34HC-1 motors in conjunction with 10:1 gear heads give a maximum driveshaft speed of 300 RPM. With 16 inch drive wheels, this equates to 14.3 MPH. Johnny 5's maximum speed is regulated to 105 RPM, or 5 MPH, in accordance with IGVC regulations. In testing, the vehicle was able to reach speeds of 5 MPH on level ground.

8.2. Ramp Climbing Ability

The Silvermax motors have a stall torque of 422 in-lb after the 10:1 gear reduction. Although, the IGVC rules specify that a vehicle should be able to transverse a 15% grade (8.5 degrees), the team specified that Johnny-5 should be able to climb a 15 degree incline. This provided a factor of safety in the case of unexpected conditions during competition. During testing, Johnny-5 was able to climb inclines of approximately 35 degrees.

8.3. Reaction Times

From initially polling the sensors to issuing a command to the motor, the software takes 0.067 seconds to complete a cycle. The sensors are able to collect and transmit data faster than the software refresh, leaving processing as the limiting reaction factor. Depending on when the obstacle is detected by the sensor, it could take between 0.067 and 0.13 seconds from the time an obstacle is sensed to when a signal is sent to the motor for the Autonomous Challenge. At a speed of 5 mph, the maximum 0.13 reaction time means that the vehicle will move 0.95 feet before the motors start reacting to an obstacle. This distance is well within the sensing range of Johnny-5.

8.4. Safety Considerations

Safety has been, and continues to be, the most important objective in designing and operating Johnny-5. The team successfully implemented safety features in the mechanical, electrical, and software systems. An important safety feature of the mechanical system is the jack stand, which is used in start up procedures and indoor testing. Electrically, Johnny-5 has both a remote controlled E-stop and vehicle mounted E-stop push button. The remote E-stop has been tested to distances of 150 feet. The on-board E-stop is located on the camera mast and is easily accessible.

9. COST ANALYSIS

Table 9.1 shows the cost to fabricate Johnny-5.

Table 9-1: Cost Breakdown of Johnny-5

Vendor	Item	Quantity	List Cost (each)	Team Cost
McMaster Carr	6063 Al Swuare Tubing 1-1/4" X 1-1/4"	84 ft	$120	$120.00
Frame Materials	Electronics Box	1	$100	$50.00
Frame Materials	Aluminum Cover	1	$300	$0
Allied Electronics	Electronic Parts	1	$211.30	$211.00
Sager	Laptop	1	$2,500.00	$0
National Instruments	Serial to PCMCIA converter	1	$495	$0
East Coasters Bike Shop	16" X 2.5" BW Tires	2	$18.00	$36.00
Northern Hydraulic	10" Pneumatic Swivel Caster 500 lb	1	$22.99	$22.99
Fairchild Semiconductor	PWR MOS UltraFET 80V/75A/0.010	5	$12.50	$0
Hawker Odyssey	12V sealed Lead-Acid Battery	2	$170.00	$0
Soneil	24V/8A Battery Charger	1	$160.00	$145.00
Skyway Machine, Inc	16" Tuffwheels with Disc Brake Hub	2	$42.50	$0
Quicksilver Controls	48V DC High Output Servo Motors and gearhead	2	$1,225.00	$2,450.00
Unibrain	Wide Angle 80.95 deg Firewire Camera	1	$81.75	$81.75
Novatel	Propack-LB DGPS	1	$7,995.00	$2,995.00
Sick	LMS-221 Laser Range Finder	1	$5,927.25	$5,927.25
Yamaha	EF 1000is Generator	1	$700.00	$700.00
Total Cost			$20,081	$12,738.99

10. CONCLUSION

Johnny-5 is an autonomous ground vehicle that was designed and fabricated by students at Virginia Tech. Johnny-5 was designed using the latest design and simulation tools, resulting in a reliable, compact, and safe system. An onboard generator can power the vehicle for up to 9 hours of continuous operation. A single powerful computer running National Instrument's LabVIEW software streamlined systems integration. We believe Johnny-5 will provide an adaptable and reliable platform for this and future competitions.

Appendix A: Electrical schematic for Johnny-5