APOLLO III: IGVC 2017 Design Report

Bluefield State College Submitted: May 15, 2017

Team Captains:

William Lambert lambert_cw@live.bluefieldstate.edu

Michael Goforth goforth_me@live.bluefieldstate.edu

Team Members:

Evan Rees rees_em@live.bluefieldstate.edu

Chris Knight knight_cp@live.bluefieldstate.edu

Samuel Mallamaci mallamaci_sj@live.bluefieldstate.edu

Ian Fields fields_im@live.bluefieldstate.edu

Daniel Seide seide_d@live.bluefieldstate.edu

David Blankenship blankenship_de@live.bluefieldstate.edu

Waleed Alosaimi alosaimi_wa@live.bluefieldstate.edu

Justin Toler toler_jl@live.bluefieldstate.edu

Samuel Stephens stephens_sr@live.bluefieldstate.edu

Jesse Edwards haynes_je@live.bluefieldstate.edu

Shonté Cargill Cargill_st@live.bluefieldstate.edu

Statement of Integrity from Dr. Robert N. Riggins:

I certify that the design and engineering of Apollo III by the 2016/2017 Bluefield State College Robotics Team has been significant and equivalent to a senior design course.

Robert N. Riggins

Dr. Robert N. Riggins

Faculty Advisor

BLUEFIELD STATE COLLEGE: APOLLO III

Michael Goforth[*], William Lambert[†], Sam Stephens[‡], Shonté Cargill & Evan Rees[§], Chris Knight[**]

Bluefield State College's Robotics Team entered Apollo III into the 25th Annual Intelligent Ground Vehicle Competition (IGVC). The objective was to design and build a functional auto-nav. robot with the ability to navigate an obstacle course the IGVC staff designed. The Robotics Team approached this task as a yearlong project following a design process that the Team agreed upon. The design process followed a cyclic pattern which this report describes. At the end of the yearlong project, the Team produced a fully functional robot that is a predicted competition contender. This report contains a full description of the robot's mechanical, electrical, and software aspects.

INTRODUCTION

Apollo III is a modified version of Bluefield State College's 2016 entrant, Apollo II. Apollo III claims many improvements over its predecessor, such as a new modular mast platform, a reconfigured electrical system, and redeveloped software. Like Apollo II, Apollo III is an intelligent vehicle that can navigate the IGVC obstacle course. To accomplish this, Apollo III comes equipped with many sensor systems, which include a wide-angle camera, a laser measurement system (LMS), a digital compass, and a GPS. These sensors interface to an internal laptop running National Instruments' LabVIEW software. Apollo III keeps the innovative fiberglass body that the BSC Team implemented on the 23rd IGVC robot, Apollo I, with the addition of the new fiberglass mast platform. This body rides atop a versatile 24 V Electric wheelchair base that has zero-degree turning capabilities. The Team anticipates that Apollo III will be a strong competitor at the 2017 IGVC.

ORGANIZATION

During the beginning of each academic year, the Robotics Team prepares for the next IGVC by organizing the Team. The first task is to recruit new Team members and introduce them to the robotics program. Once everyone is familiar with the robot and the robotics program, the BSC Team decides how to proceed through the two semesters before June. The fall semester is primarily a teaching semester. The Robotics Team recommends that new robotics students take an introductory robotics course that the college offers. In this course, they learn the programming methods they need to work on the robot as well as mechanical and electrical skills they need for practical work. During this time, returning students work on new design aspects for the robot. The spring semester is when the Team performs major work on the robot through a special design course. Implementing new designs as well as performing preemptive testing is the primary focus before the competition.

[*] Team Captain and Report Author, Mechanical/Electrical Engineering, Goforth_me@live.bluefieldstate.edu

[†] Team Captain, Mechanical/Electrical Engineering, Lambert_cw@live.bluefieldstate.edu

[‡] Mechanical Lead, Mechanical Engineering, Stephens_sr@live.bluefieldstate.edu

[§] Co-Electrical Lead, Electrical Engineering, Cargill_st@live.bluefieldstate.edu

Co-Electrical Lead, Electrical Engineering, Rees_em@live.bluefieldstate.edu

[**] Software Lead, Computer Science, Knight_cp@live.bluefieldstate.edu

DESIGN: ASSUMPTIONS, PROCESS, AND COST BREAKDOWN

Assumptions:

At the start of Apollo III's design process, the Robotics Team made a few key assumptions. It assumed that the course terrain would be relatively smooth with slight elevation changes consistent with that of a field. The Team equipped Apollo III to operate in light rain and other mild weather conditions. Lastly, the Team assumed that the old rules would be like the new ones so that it could start the design process early.

Design Process:

Table 1 shows the design Team for Apollo III. Figure 1 illustrates a 7-step cyclic strategy for designing the robot. The seven steps are the following: identify the problem, prioritize, delegate Team members, research solutions, design a solution, implement the design, and then test the solution. The first step, to identify the problem, occurs during the return trip from the IGVC. The Team analyzes the problems that it faced during the competition and brainstorms solutions to these problems. The second step, to prioritize, involves ranking problems on the basis of their impact on overall performance. The third step is to delegate Team members to the highest-ranking issues. The number of students that the Team assigns to each concern varies depending on the issue's rank, the time the task requires, and the skillsets of the available students. Table 1 displays the Teams' majors, which relate to the students' individual credentials. After each design Team has an assigned issue, the Team can advance to the research phase of the process, the fourth step. This step involves researching feasible solutions to the individual problems and determining that the Team has the most practical answer. Then, in the fifth step, the students work on a suitable design for Apollo III. The sixth step is to implement this design on the robot and ensure that everything is operational before testing. In the last step, the BSC Team rigorously tests the robot to confirm that the prototype works correctly and effectively. If the test results are not optimal, then the group restarts the seven-step cycle.

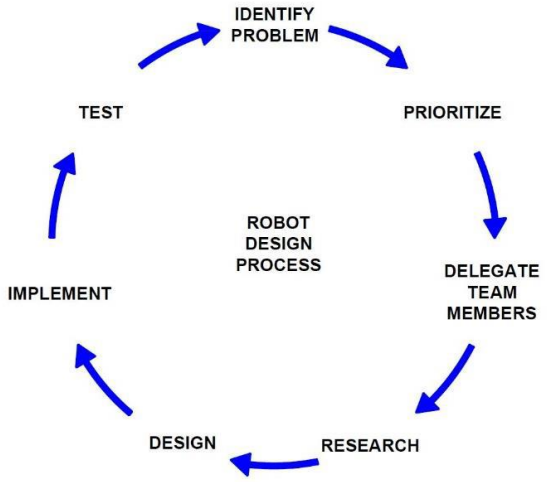

Figure 1: Design Process for Apollo III

Table 1: IGVC Team Member Information

Name	Major	Status	Hours
William Lambert	Mechanical/Electrical Engr.	Captain	100
Michael Goforth	Mechanical/Electrical Engr.	Captain	50
Samuel Stephens	Mechanical Engr.	Mechanical Lead	10
Chris Knight	Computer Science	Software Lead	35
Shonté Cargill	Electrical Engr.	Electrical Lead	17
Ian Fields	Mechanical/Electrical Engr.	Member	2
David Blankenship	Mechanical/Electrical Engr.	Member	2
Samuel Mallamaci	Mechanical Engr.	Member	20
Jesse Edwards	Mechanical Engr.	Member	5

Evan Rees	Electrical Engr.	Member	20
Justin Toler	Electrical Engr.	Member	15
Waleed Alosaimi	Electrical Engr.	Member	15
Daniel Seide	Electrical Engr.	Member	5
		Total:	296

Cost Breakdown

Table 2 organizes the cost breakdown of Apollo III's components by item. If the need to reproduce the robot arises, the grand total at the bottom of this table is a good estimate of what it would cost, excluding labor and contracted work. The costliest items are the sensors, laptop, and the electric wheelchair bottom. Other items such as the motor controller, wireless electrical devices, and the body are relatively cheap. The last item, miscellaneous, represents all minor costs that accumulate over time. Some of these costs are electrical conductors; small electrical components such as resistors, IC chips, etc.; lights; switches; and related objects.

Table 2: Cost Breakdown by Item

Item	Cost
Hokuyo LMS	$5,600
Dell Precision 7510	$2,580
Hemisphere GPS	$2,400
Basler USB 3.0 Camera + Lens	$1,800
Jazzy Wheel Chair Bottom	$1,031
Body Fabrication	$750
Maretron Solid State Compass	$600
2 Yellow Top 12V Batteries	$240
SaberTooth Motor Controller	$180
2 XBees and Antennae	$120
Wireless Router	$80
3D Printed Components	$20
Miscellaneous	$1,100
Grand Total	$16,501

INNOVATIONS

Software Modularity:

Apollo III's LabVIEW software is modular in the sense that replaceable program blocks, Virtual Instruments (VI's), compose all software functions. Apollo's software improves this inherent modularity via an innovative Global Information Cluster (GIC). This GIC contains all known information about Apollo III and is available to all VI's. By using the GIC, the programmer can switch easily between VI's to suit different programming scenarios. For example, the programmer can readily exchange a local navigation algorithm for a global navigation algorithm or add new sensor VI's without affecting the rest of the software. This innovative concept can apply both to a priori programming and to adaptive real time operations.

With the inclusion of the DLL (Dynamic Link Library), Apollo III reaches an even higher level of modularity than it would with the GIC alone. The DLL allows programming in any language such as C/C++. LabVIEW works well for connecting instrumentation but is cumbersome and inefficient for more complex algorithms. C++ programs can be compiled to very fast, efficient machine codes for more intensive image processing and pathfinding algorithms. It is not necessary to change the LabVIEW code to improve and recompile these C language programs.

Trap Escape:

Apollo III employs a student-designed algorithm that creates a global map as it navigates through the IGVC course. This special algorithm prevents Apollo from going into the same trap more than once. The Software Design section describes this algorithm in more detail. Previous renditions of Apollo have entered endless loops, easily trapping themselves. Now, Apollo III's innovative algorithm makes Apollo III less susceptible to traps than previous Apollo versions.

Weather Capabilities:

Apollo III has many features that lend well to its ability to function in different weather conditions. Among these features are the following: white paint with a clear gloss finish; curved body shape; an internal, static pressure fan; and weather stripping. The white paint and clear gloss finish allow Apollo to operate in hot and sunny environments because these features reflect solar radiation, keeping the internal temperature of the body at acceptable levels. In rain, the curved body shape allows water to roll off the robot to the ground. This aspect and the positive pressure fan, which blows air around the seams in the weather stripping, allow Apollo to operate in rain. The combination of these features allows Apollo to function in a large variety of weather conditions.

MECHANICAL DESIGN

Overview:

Apollo III's mechanical design focuses on being lightweight, modular, highly maneuverable, and easy to assemble and disassemble. Apollo III consists of two main integrated mechanical fabrications, the drive chassis and the body. The Team altered the electric wheelchair bottom of the drive chassis so that it can reach high speeds and perform zero-degree turns. Wood and fiberglass form the body, giving it lightweight, strong, and weather resistant properties. Overall, the key components of Apollo III's mechanical design encompass simplicity without compromising performance.

Figure 2: Stages of Body Development

Structural Design, Frame, and Suspension:

The students designed and fabricated Apollo III's body out of wood and fiberglass. It is strong and light-weight, and its modularity allows for easy assembly and disassembly. Figure 2 shows Apollo III's body during various stages of the structure's development. The structure has a curved shape, which makes it safer by not having any sharp edges while adding to its aesthetic appeal. A professional automotive white paint and clear gloss finish coat the body. These coatings aid in the reflection of solar radiation. This helps keep all our electrical components cool during operation.

With a light-weight body, the motors do not have to work as hard to sustain full mobility. This light-weight form also reduces drain on the batteries, allowing Apollo III to run for hours. The table in the Performance Testing and Assessment section gives information on battery life. The BSC Team can effortlessly remove the top half of the body's cabin without tools, providing access to all of Apollo III's internal components. The body also has a modular design so that Team members can add or remove parts depending on situational demands. For example, the mast connects to the body via a simple internally reinforced PVC pipe connection that members of

the Team can easily exchange for another mast design or a different component altogether. This is one example of the many ways in which Apollo III's design allows for flexibility in the field.

Figure 3 shows Apollo III's drive chassis, a Jazzy electric sports wheelchair base that the students have further developed. This model allows Apollo III to achieve the upper limits of competition speed without sacrificing maneuverability. Also, the suspension allows for traversal of various terrains and environments. The students modified the frame to accommodate a battery tray that they redesigned and built. This tray holds two 12 V batteries and slides out of the chassis for easy access. A payload slot, accommodating 20 pounds, is an addition to the front of the frame.

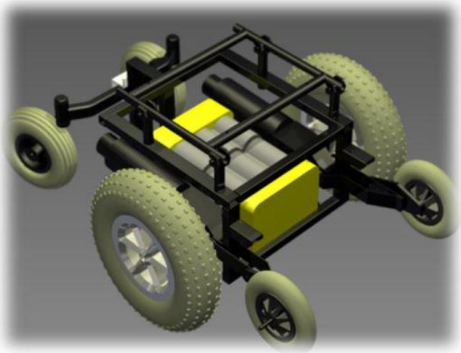

Figure 3: Apollo III's Drive Chassis

Cooling:

Apollo III must operate in the sun for extended periods of time, so it is critical that the internal components remain cool. Apollo III's body has features to meet this goal. Its white color and reflective finish redirect radiation away from the robot. The BSC Team has also installed a high static pressure fan in the underside of the body. This fan circulates air around the laptop, electrical components, and sensor systems to aid in cooling.

Weather Resistance:

The design of Apollo III allows it to operate in various weather conditions. To meet this end, Apollo III's body possesses a curved shape. This curved shape causes water to roll off the robot and away from areas susceptible to moisture damage. The fan additively discourages water entry by driving air between every seam in the body. A minimal number of parts compose the body, but where there are assembly points, weather stripping is present.

ELECTRICAL DESIGN

Overview:

Because of Apollo II's outstanding performance at the IGVC 2016, the Team decided not to change most of Apollo II's electrical system for Apollo III. Originally, the students removed the joystick and control module of the wheelchair control system from Apollo I and integrated a completely student-built design into Apollo II. The major changes from Apollo II to Apollo III were safety improvements. Apollo III, unlike its predecessors, has a light notification system and a router for peer-to-peer communications. These systems relay the robot's processing to the users who are both in the field and at the control hub respectively. The control system consists of an XBee wireless transceiver, Parallax Propeller 8-core microprocessor, and a Sabertooth 2x60 motor controller. A full suite of sensors feed data to an onboard laptop. Various safety features are also an important part of the overall electrical system. The focus of this design is on safety, reduction of power consumption, and fully customizable dynamics control.

Power Distribution System:

A solitary source powers all of Apollo III's components. This solitary source is two Optima Yellow-Top 12 V DC batteries which are in series to produce 24 V DC. The 24 V DC power supplies the 24-to-12 V DC-to-DC converter; it also supplies the motors and brakes through the Sabertooth motor controller. The 12 V DC output of the converter supplies all the other components

on Apollo III. Having a solitary source for everything, coupled with an easy "plug and charge" system makes charging Apollo III simple. A new power distribution unit with switches aids in power savings using an integrated switch panel for all sensors and devices. These switches allow users to disable quickly unneeded sensors for testing.

Electronic Control System:

The Mechanical Design section mentioned that Apollo III's chassis was originally an electric wheelchair. As such, the original control system was the wheelchair controller, complete with its own joystick and control circuitry. One of the more demanding design challenges of the past was to completely overhaul Apollo's control system in 2015-16 to a completely student-built and design-specific system. The students made a new control system that incorporates the Parallax Propeller 8-core microprocessor coupled with a Sabertooth Motor controller. They put extra consideration into transient suppression and electrical isolation of parts to prevent future component failure. The BSC Team used opto-isolators between the motor controller and microprocessor to physically separate the two systems since they operate on distinct power levels. The Kangaroo module, from Dimension Engineering, creates a feedback loop with the motors. The Team added a ferrite core toroid to the motor wiring harness to suppress transients, specifically reverse voltage of the motors. The brake relay uses a flyback diode to reduce transients, a capacitor to reduce arcing, and an opto-isolator ensures operation of the brakes while keeping the higher voltage isolated from the microprocessor. Figure 4 depicts Apollo III's electrical system.

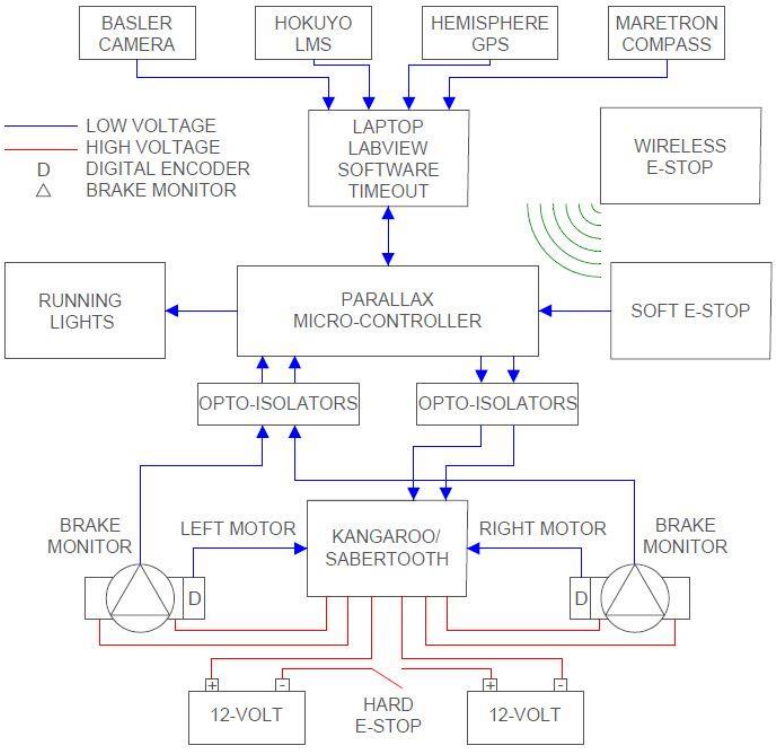

Figure 4: Apollo III's Electrical System Illustrating Sensor Inputs, Processing Logic Flow, and the Separation of the Power System and the Control System via Opto-Isolators

Sensor Systems:

Apollo III uses the following four sensors: a LMS, a camera, a GPS receiver, and a compass. These sensors interpret information from the outside world. The proven accuracy and speed of these sensors contribute to Apollo III's ideal performance. The speed of these devices ultimately determines the overall cycle time of the software. The BSC Team specified that a sensor should update once every 50 ms at a minimum, because at a speed of 5 mph the robot will travel approximately 4 inches per cycle. The Team found this to be an acceptable speed. The following bullet points describe the sensors:

- Camera: Figure 5 shows the Basler USB 3.0 outdoor camera. The camera has many favorable qualities. With a frame rate of 90 frames per second and high-speed data transfer, this camera provides more than enough speed for the robot. Additionally, automatic white balancing, gain adjustment, and shutter speed control allow for excellent vision stability in any lighting condition. When combined with a horizontal 125-degree field-of-view lens that has only 3% distortion, the Apollo III vision system has excellent precision and versatility.

Figure 5: The Basler Camera

- Laser Measurement System (LMS): Figure 6 displays the Hokuyo LMS for object detection that Apollo III utilizes. With a 270-degree field-of-view measured in 0.25-degree increments and a detection range of up to 30 meters, the LMS provides extremely accurate object detection. It cycles at 40 Hz, allowing ample time for Apollo III to detect any obstacles in its path. The LMS also features data clustering, specular measurement, and adjustable resolution levels for maximum customization.


Figure 6: The Hokuyo LMS

- GPS: To obtain positioning data, Apollo III uses a Hemisphere A21 GPS (L1, GNSS, L-Brand) from Blueplanet Geomatics, which Figure 7 illustrates. This provides position, direction, and speed data, allowing Apollo III to track both its own position and those of user-defined waypoints. The A21 GPS houses two antennae in the same location; these antennae make differential corrections. This GPS unit runs at 20 Hz, making it easy for Apollo III to navigate through the course at high speed.


Figure 7: The Hemisphere GPS

- Compass: Figure 8 shows a Maretron Solid State Compass which assists in determining vehicle heading. A compass complements the GPS since the GPS provides less accurate data when it is stationary or moving slowly than when it is moving quickly. The compass provides an accuracy of 0.1 degrees, and updates at 100 Hz to verify our direction. The best performance is acquired from the GPS and compass when they are work in tandem. The Maretron compass can measure pitch and roll up to 45 degrees, preserving its functionality on inclines.

Figure 8: The Maretron Compass

Safety Features:

Apollo III has many safety features. For example, we have four independent ways to stop the robot quickly and effectively: two physical switches, a combination of software timeouts, and a wireless emergency stop. The two physical switches reside on the body of the robot itself. The soft E-stop gives the processor a software pause command, and the hard E-stop physically cuts power to all motors and sensors. The wireless E-stop sends stop commands, well exceeding the minimum range requirement of 100 ft. Also, there is a firmware timeout should the Sabertooth fail to receive, within 100 milliseconds, a valid signal that stops the robot. Another safety feature the Sabertooth provides is the ability to limit the maximum speed. Apollo III also contains a separate brake/motor monitor circuit. If the brakes and motors fail to synchronize properly, then multi-core Propeller microcontroller commands the robot to stop. One of the Propeller processor's independent cores controls the brake/motor monitors. Figure 4 shows these safety features.

SOFTWARE DESIGN

Overview:

Students use LabVIEW to develop Apollo III's software. Using LabVIEW allows Apollo's programmers to create easily a sophisticated graphical user interface that makes it simple to monitor data, change settings, modify code, and debug. LabVIEW uses a visual programming environment, which allows new programmers to be involved in writing code along with the experienced programmers.

In the 2016 IGVC, Apollo II's path planning algorithm revolved around mapping sensor data to a "local map", an 80 x 80 2D grid of weighted nodes, that represented the area of two to six meters around the robot. After every cycle, Apollo II refreshed its "local map," discarding all its prior data. Apollo III in the 2017 IGVC creates a much larger global map, remembering node attributes as it traverses the course. Apollo III can switch between these navigation methods before and/or during the run.

The LMS locates obstacles, and the vision system detects lines. These sensors send their information to an onboard laptop which maps the data to the grid of nodes. The GPS and compass send location and heading data to the grid, where the obstacle data is, to select a goal on the map that will progress the robot to the next waypoint while still avoiding obstacles. The path planner will then create a safe path from the robot to the goal using a weighted lowest cost path equation. The last step is to smooth the path, which determines the commanded heading and speed of the robot.

Obstacle Detection and Avoidance:

Apollo detects objects with its laser measurement system. The LMS provides an array of distances to detected objects within a 270º arc that is centered directly in front of the robot. Each node on the global map represents a 0.1 meter square on the real course. Apollo III's LMS software assigns all objects the LMS detects to specific sets of nodes.

Apollo III avoids obstacles while moving toward the next waypoint by using the methods that the Software Strategy and Path Planning and the following subsections describe below. The ultimate result of the pathfinder is an initial vector that Apollo III follows toward the waypoint while moving around obstacles.

Lane Following:

Apollo III uses a camera to detect white lines on the ground. Image processing reduces the camera input to a 492 X 658 binary bitmap consisting of pixels. A one represents white and a zero represents anything else. As with obstacle avoidance, Apollo III's vision software assigns the detected white lines to specific nodes on the global map. The pathfinding algorithm treats the detected lines and obstacles as nodes to avoid.

Waypoint Navigation:

GPS and Compass provide the latitude, longitude, and heading of Apollo III to the global map. Apollo III's software places Apollo III on the global map by assigning a node and heading to the robot. Waypoints have assigned latitudes and longitudes as well, so the software also assigns nodes to all the waypoints. Therefore, the global map initially contains the robot node and all the waypoint nodes. Because obstacles and paths exist between the robot and waypoints, Apollo III's software must follow various methods to reach the next waypoint.

Software Strategy and Path Planning:

The strategy Apollo III uses is to place all waypoints and initial robot information on the global map before run-time. Then, during run-time, the software begins updating robot location and begins placing LMS and vision data on the global map. Apollo III then selects a goal. If a known path exists between robot and the next waypoint, then the goal is simply the waypoint. If no known path exists between robot and waypoint; due to unexplored nodes, known obstacles, or known lanes; then the algorithm assigns a "non-waypoint" node as a temporary goal. Finally, Apollo employs a path smoothing algorithm to produce a direction and speed to send to the motor controller.

Apollo's earlier pathfinding algorithms used the previously mentioned 80 x 80 2D array of weighted nodes. Various algorithms were utilized to implement this system however all were susceptible to infinite loops because when the robot was presented the same data repetitively, it would make the same decisions. Pathfinding algorithms; such as Dijkstra's algorithm, A*, and Jump Point Search; rely on a complete map to find an optimal path. Since Apollo does not have access to a complete map of its surroundings, Apollo III uses a new algorithm to find its path.

Map Generation:

In previous generations of Apollo, the 80 x 80 map was discarded after every cycle. This left the pathfinding algorithm to choose a path based solely on the most recent batch of data reported by its instrumentation. Apollo III now uses a new global mapping system to store persistent data on a much larger (2048 x 2048) two-dimensional array of nodes. These nodes contain location variables for obsticles detected by the LMS or the camera. To improve accuracy, information mapped at long distance is replaced by more reliable data obtained at a closer proximity. Within a certain distance threshold, nodes will not be overwritten with new data.

Goal Selection and Path Generation:

The new pathfinding algorithm looks for the current waypoint on the map. Like A*, it uses a priority queue to attempt to quickly find the optimal path without evaluating unnecessary nodes. However, if the current waypoint is not on the map, then Apollo must cross unmapped nodes to reach the waypoint. The new algorithm finds nodes adjacent to unmapped nodes during its search for the waypoint. It evaluates each of these special nodes, and it stores the best of the nodes based on the heuristic assigned to that node and its distance from the robot. The priority queue is analyzed completely if the waypoint is not found, leaving Apollo with a destination adjacent to an unexplored area. If the waypoint is located, then the waypoint node will become the destination. As the path-finding algorithm traverses the map, it assigns each node a parent node. This provides Apollo a path back to the robot from the destination. A simple linked-list traversal back through the path to Apollo yields a path of nodes which can then be smoothed to generate a direction and speed. These values are converted to PWM signals which are then sent to the motor controllers.

Figure 9 is a simplified graphical representation of Apollo's pathfinding algorithm. The white circle in the center of the map represent the robot, the red nodes represent physical obstacles or white lines, and the pink nodes represent an adjustable buffer zone to prevent Apollo from hitting objects or crossing lines since the robot occupies more than one node.

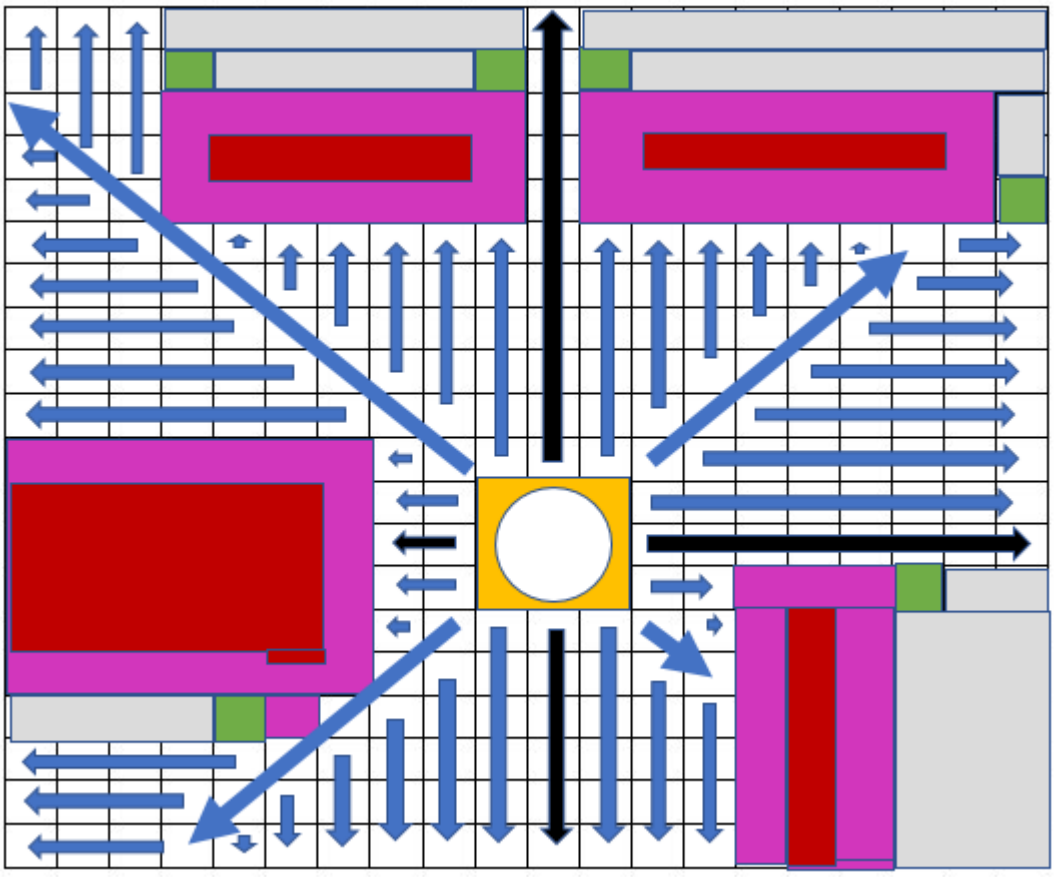

Figure 9: A Graphical Representation of the Pathfinding Algorithm

The algorithm begins by inserting the eight nodes adjacent to Apollo into a priority queue if they are clear of obstructions. Those initial nodes are colored orange in the illustration. Then, using a heuristic weight based on distance from the waypoint and total path length from Apollo, the nodes are evaluated from the queue. The black arrows on the map indicate nodes which are directly adjacent to Apollo. The nodes along a diagonal vector require that each horizontal and vertical component of that vector be explored. Nodes which cannot be located by the horizontal or vertical components of a diagonal vector are considered a blocked node and are inserted into the queue as a node of interest. In Figure 9, these blocked nodes are colored green. Once an obstacle or the edge of the map is encountered, the next node in the queue is evaluated. As the algorithm operates it saves time by overlooking uninteresting nodes (those passed by the horizontal and vertical components of the diagonal vectors) in search of the waypoint.

Insertion into the queue requires valuable time. The sample map in Figure 9 contains 400 nodes, but by only inserting 6 of those nodes into the queue Apollo can conserve valuable processing time. The algorithm would not necessarily evaluate all 8 of the initial seed nodes first since the green nodes would be inserted into the priority queue based on their heuristics. The gray areas remain as unexplored areas until the green blocked nodes are evaluated from the queue. After these blocked nodes are evaluated the grey nodes are considered. This pattern of evaluating blocked nodes and moving on continues while the robot navigates an area in search of a waypoint.

Additional Creative Concepts:

The Team is in the process of developing a few additional software concepts. These concepts include an adaptive weight system that permits Apollo to alter the nodes' values and an adaptive exception-handling algorithm that gives Apollo the ability to choose between different navigation algorithms depending on which one will produce the best results. Traps are an example of a situation in which adaptive weights benefit the robot. If a GPS waypoint is pulling the robot into a trap, then Apollo gives less weight to GPS data allowing the robot to escape. Switching between distinct types of obstacles is an example of when exception-handling is useful. If the robot is using a "line-follower" algorithm when it meets a great expanse, then the robot can switch to an algorithm that is suitable to open spaces.

JAUS

Overview:

The goal is to achieve three levels of JAUS interoperability. The first level is to connect JAUS as a subsystem of a parent system. At IGVC, the judges provide the system. In the future, BSC will develop other subsystems which can intercommunicate with Apollo's system. Implementing JAUS allows Apollo to join any other system which conforms to the JAUS protocol. The second level of interoperability provides communication between Apollo's subsystem and its nodes, and the third level of interoperability requires low level programming to implement JAUS within individual nodes.

Level 1:

Apollo's computer employs a wireless adapter to receive Wi-Fi transmissions and an Ethernet port for wired access. Because of a new router, Apollo can provide its own network. This allows Apollo to implement the first level of operability by sending and receiving JAUS messages to its parent system. The recently added DLL allows users to send a string to a C++ function, which parses the string using the extensive C++ string library. In the process the parent system receives this string. A text file can easily store these strings with a timestamp to document all communication with the parent system. Another function processes and stores outgoing messages.

Level 2:

Once it parses a message, the subsystem sends a query or a command to a corresponding node. Each node has handlers to comply with the JAUS directives it receives. The Level 1 parser provides only the substring of the JAUS message that the specific node needs. Each node further parses the string and creates messages for each of its components. Finally, the nodes return messages to only the components that must answer the query or comply with the command.

Level 3:

At the third level, components receive queries and commands as a JAUS string and respond by following any commands and transmitting a response to their parent nodes. The implementation at this level varies greatly between components. Some components simply perform commands and send an acknowledgement while other nodes construct a string from values obtained from instrumentation. Messages sent back to parent nodes will be used to construct strings to be sent back to the subsystem which will in turn construct a response to send to the parent system.

FAILURE MODES, POINTS, AND RESOLUTIONS

Vehicle Failure Modes and Resolutions/Strategies:

Failure:	Response/Resolution Strategy:
A. Unnecessary stops on the course due to a lack of a method for trap escape.	a. Redesign of the software to include a memory of past locations.
B. Turning around on the course and going backwards caused by endless loops.	b. Adjustment of the weights set on GPS data and using the memory mentioned above Possible Dynamic Adjustment.
C. Vision processing creating a bottleneck on other processes slowing down cycle speed.	c. Upgrade our onboard laptop to one with higher capabilities, eliminating the bottleneck on vision.

Vehicle Failure Points and Resolutions/Strategies:

Failure:	Response/Resolution Strategy:
A. Less than ideal wiring in the lower power system creating the potential for a ground fault.	a) Rewire job of key areas and inspection of existing wiring.
B. Magnetic field from a ferrous frame in the body causing interference with the compass and other components.	b) Fabrication of a new aluminum frame to fully eliminate the magnetic field produced by the ferrous metal.
C. Stress fractures in the fiberglass body – stress concentrators	c) Utilization of rubber grommets to more evenly distribute loading.
D. Inaccurate movement and no way to adjust/compensate.	d) Integrate a feedback loop from the motors using encoders.
E. Center of gravity too far forward after the addition of the new mast platform.	e) Addition of a counter balance to place the center of gravity over the main wheels. Make certain that max weight capacity of 500 lb is not exceeded for the base.

SIMULATIONS

Figure 10 displays Apollo III's simulation program that allows users to create their own courses and test how Apollo III's software reacts to different situations. The Robotics Team has created several different courses to test Apollo III's software. This has proved to be invaluable to software development because the BSC Team does not have the means to create complicated courses on campus that would allow practical testing. It is also much easier and faster to create a virtual course than to create a real course. Simulations are critical to the development of Apollo III during the vehicle's downtime when the students are redesigning physical aspects.

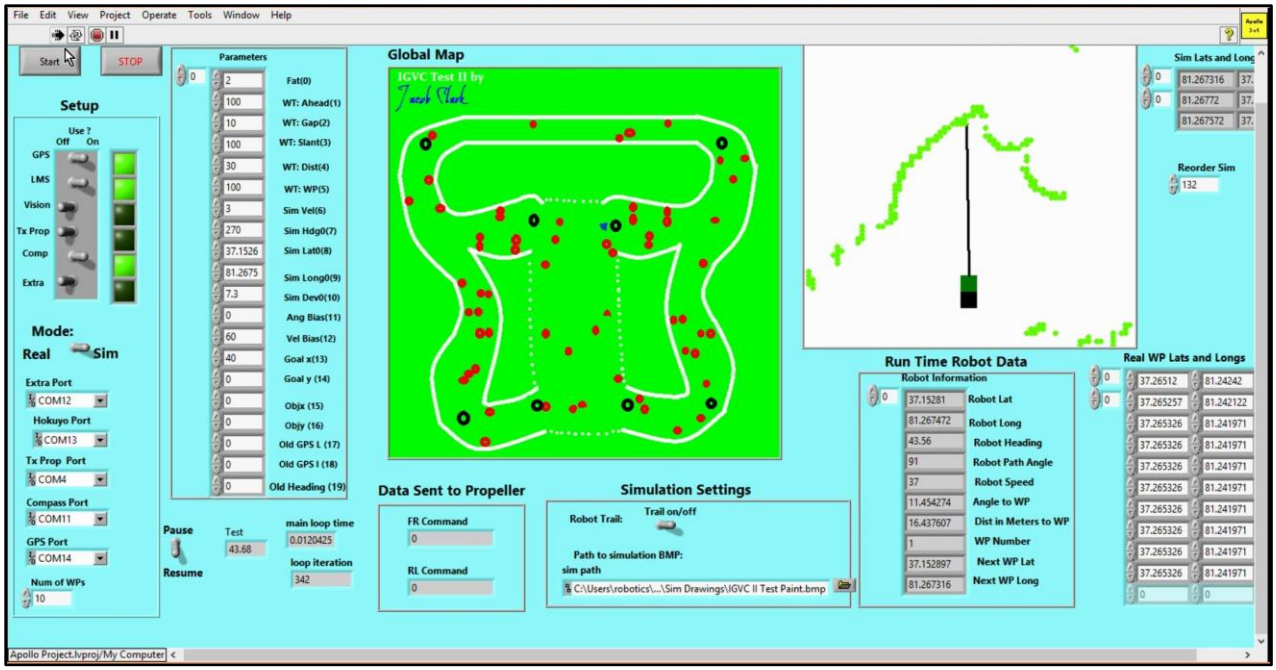

Figure 10: Front Panel View of the Apollo III Simulation VI. Both the Local (left) and Global Maps (right) are Shown.

PERFORMANCE TESTING AND ASSESSMENT

The BSC Team begins the process of identifying failure points and modes immediately after each IGVC. The process begins with an in-depth analysis of the performance from the competition by the Team. The Team considers anything that does not perform at or above expected levels to be a failure point. After identification of all failure points in each area, the Team identifies the cause of each failure point. If a factor other than the robot's design and programming causes the failure, then the Team considers multiple resolutions. Team members redesign or replace aspects of the robot that function poorly or experience a design failure. After the Team identifies all failure points and resolutions, research and analysis of possible resolutions begin. The Team selects improvements based upon the ability of the resolution to benefit the functionality of the design and its applicability to the project's design philosophy.

Apollo III avoids complex traps by utilizing adjustable variable weights for path-planning as well as a global mapping system. While Apollo III moves toward the goal, the global mapping system is constantly exploring and expanding the known areas of the map. It is also storing information about each node in the global map. If the selected goal pulls Apollo III into an unpassable or unfavorable area, then the pathfinding algorithm identifies this as a trap and seeks an alternate route to the waypoint. Once the robot achieves the waypoint, the path-planning algorithm can select a new goal that puts it on course toward the next waypoint using the data the global mapping system stores. Because the global mapping algorithm has stored the information regarding the traps and the obstacles it encounters, these traps and obsticles should no longer ensnare Apollo in infinite loops. Apollo III will only return to an area it has identified as a trap if this area becomes a waypoint.

Table 3: This Table Shows the Categories of Tested Robot Performance. Actual Performance is Based on Empirical Gathering of Field Data.

Category	Predicted Performance	Actual Performance
Reaction Time	50 ms	30 - 70 ms
Battery Life	N/A	2.75 - 5.50 hr
Obstacle Detection Distance	30 m	50 m
GPS Waypoint Arrival Accuracy	2 ft. 67% of the Time	2 ft. 67% of the Time
Ramp Climbing	N/A	30% Grade
Speed	5 mph	< 5 mph

The initial assessment of Apollo III bodes well for the competition in June. The mechanical design is working properly with no sign of wear. The electrical systems are functioning and have passed the test of time since last IGVC. Apollo III's software surpasses that of Apollo II, and at the time of this report, Bluefield State College continues to test and optimize the newest software. Apollo III is looking good, and as BSC approaches the IGVC, the Team will continue to make beneficial changes to the robot. Bluefield State College believes that it will be a contender in this year's competition.

For the future, BSC wishes to continue refining the Apollo robot. The BSC Robotics Team feels that it has a sturdy foundation upon which to build. The Team is eager to integrate adaptive exception handling into the software design so that Apollo has a more dynamic and changing algorithm based on specific scenarios that the robot finds. Also, the Team desires to utilize an inertial measurement unit into the design for even more accurate movements. Another desire is to design a standard power tie-in bus for quick and easy integration of new sensors and devices. These are a few of the improvements that are in the plan for Apollo's future.