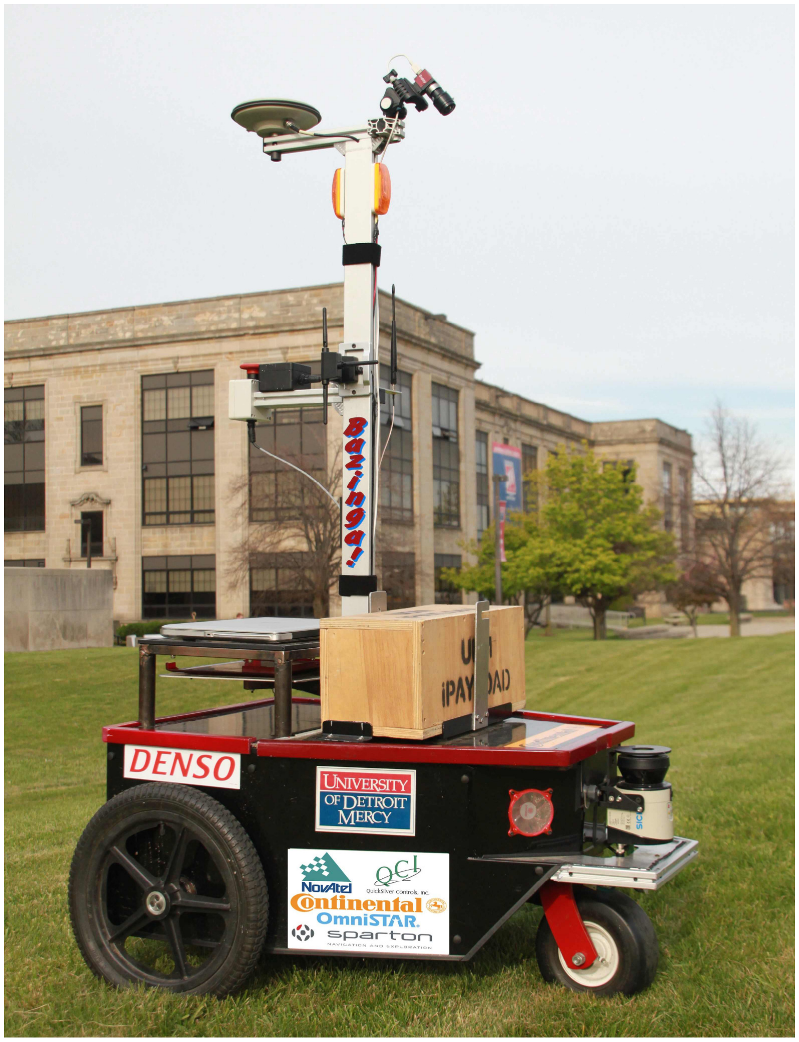
The University of Detroit Mercy
Presents

BAZINGA!

IGVC 2012 Design Report



Team Members

Peter Sutherland
Herta Llusho
Stephanie Musante
Michael Nagrant
Xingzhong Zhang
Bo Cui
Eyad Zieno
Yu-Ting Wu

Faculty Advisors

Dr. Chaomin Luo
Prof. Utayba Mohammad
Dr. Mohan Krishnan
Dr. Mark Paulik

Consultant

Cheng-Lung Lee

We certify that the engineering design in this vehicle undertaken by the student team, consisting of undergraduate and graduate students, is significant and qualifies for course credits in senior design and in the Master's program respectively.

____

Prof. Mohammad, Dr. Luo, Dr. Krishnan, Dr. Paulik

1. INTRODUCTION

The University of Detroit Mercy (UDM) 2012 IGVC team is pleased to reintroduce *Bazinga!*, an enhanced version of *Cerberus,* the differential drive vehicle first introduced in the 2010 IGVC. *Bazinga!,* based on a proven platform, introduces significant innovations, ranging from extensive software enhancements to an entirely new power management system. The goal for this year's IGVC team was to enhance the previous design and develop and implement solutions to new competition requirements. The resulting vehicle not only meets the additional requirements of the competition, but is more capable and robust.

2. DESIGN INNOVATIONS

The innovations incorporated in *Bazinga!* are listed below and they are further explained in detail in Section 7:

- A new self-contained charging station with associated software for the robot to locate and dock with it.
- A new 48 V power train with the associated charging system.
- A new local navigation algorithm with optimal curvature-based motion.
- A new spline-based speed modulation strategy for the Auto-Nav and JAUS challenges.
- A new flag detection addition to the vision algorithm suite and an associated dedicated Delaunay-based flag-guided navigation algorithm.

3. PROJECT MANAGEMENT

3.1 Design Process

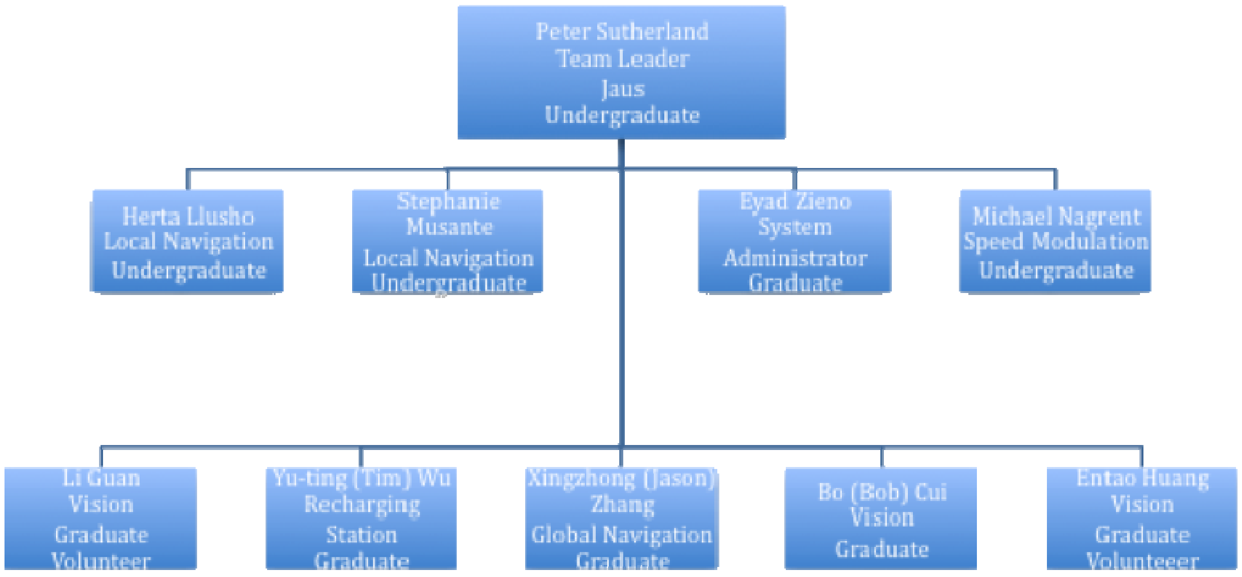
The team wanted to approach the project in an organized manner that facilitated continuous communication between all team members, allowing for sharing of key information for the various sub-tasks to be completed and identifying and resolving project bottlenecks in a timely manner. In order to accomplish this, an adaptive project framework was used that could accommodate variable scope for the individual project tasks. This is an iterative project management framework that seeks to optimize performance for each task in the design-implementation cycle.

Early in the Winter Term, a literature search was performed to find potential solutions for modifications and/or improvements necessitated by competition rule changes, as well as perceived shortcomings of our vehicle from the previous entry. A detailed feasibility study of these possible solutions and an initial design process was undertaken. Weekly deadlines were setup as part of the design process for various individual requirements. This included biweekly meetings for reporting and biweekly individual written reports, as well as individual weekly progress reports. At the biweekly meetings, the team assessed individual difficulties in order to be able to direct resources where necessary to facilitate the removal of bottlenecks. The iterative project management strategy that was used optimized performance in every task undertaken by members of the team and ensured continuous progress.

3.2 Team Composition

The team comprises 4 Senior Electrical Engineering undergraduate students and 4 graduate Electrical Engineering students. Individual assignment of tasks was based upon personal expertise and interest in topic areas.

Figure 1 indicates the existing team structure and the assumed responsibilities by each team member. The total time dedicated to the development and implementation of this project was approximately 3000 hours.



3.3 Design Objectives

In accordance with the principle of continuous improvement, the enhancements incorporated into *Bazinga!* were based on IGVC performance in the previous competition year. Both faculty advisors and team members met to review our previous vehicle's performance at the competition and that of other teams. The document generated at that meeting served as the basis for the development of the design objectives for *Bazinga!*. These were:

1. Meet all IGVC requirements.
2. Enhance the vision strategy to deal with the newly incorporated flag detection requirements.
3. Develop a dedicated flag-guided navigation algorithm.
4. Develop speed modulation techniques to optimize the course completion time.
5. Design a charging station with the associated software for the robot to locate and dock with it.

3.4 Cost Summary

Despite the fact that Bazinga! was a developmental vehicle, it was constructed while keeping cost-effectiveness in mind. The systems incorporated were either designed and built in-house or specifically purchased to best suit the project objectives in terms of functionality, performance, cost, and development time. Table 1 shows the approximate retail cost of the various sub-systems in Bazinga! as well as the team cost (some items were donated, reduced in price, or used from an earlier project).

4. MECHANICAL SYSTEM DESIGN

Bazinga! has the same mechanical design as the previous vehicle *Cerberus*. It is a 3-wheel vehicle with two rear driving wheels and a leading caster. *Bazinga!* has a width of 0.8 meters, a length of 1.04 meters, and a height of 1.8 meters. The total weight of the vehicle, not including the payload, is 110 kilograms. The frame is made out of 20 mm square steel tube stock and pre-painted 6.35 mm Alumilite sheets. This material has the same weight as 1 mm thick aluminum but is 50 times stronger, adding robustness while minimizing the vehicle's weight.

Table 1: Cost Summary

Description	**Retail Cost**	**Team Cost**	**Comments**
Frame/Body	$ 746	$746	Volunteer Work Involved
Drive Train	$ 2,471	$ 2,471	Purchased
Rear Wheels (2) & Front Caster	$ 600	$ 0	
Batteries (4) & Chargers (2)	$ 340	$ 340	Purchased
Power PCB	$ 1,100	$ 1,100	Designed In-House
Remote PCB	$ 304	$ 104	Transceiver Donated By Aerocomm
Camera, Lens, Adapter	$ 937	$ 898	
LIDAR	$ 5,500	$ 5,000	Purchased
DGPS & Antenna	$ 6,000	$ 3,500	
Digital Compass	$ 1,350	$ 0	Donated By Spartan
MacBook Computers (2)	$ 5,000	$ 5,000	
OmniStar HP	$ 1,250	$ 0	Donated
Charging Station	$ 600	$ 600	
DC/DC Converter	$ 400	$ 0	Donated By AxioMatic
TOTAL	**$ 26,598**	**$19,759**	**Savings = $ 6,839**

Bazinga! has a simple, yet solid drive train configuration (see Figure 2). It features a front caster and two rear drive wheels. Our two side-by-side 48V motors provide a continuous stall torque of 4.77 N-m. The motor's output is coupled with a 12:1 planetary servo gearhead, which has built-in bearings rated at 1400 N for radial loading and 3000 N for axial loading. It also features an output flange instead of a shaft, which takes away the need for an alignment coupling, along with two tapered roller bearings that would be required to allow for radial and axial loads. Torque is transmitted from the gearhead to the wheel through a two-part connector; it consists of a custom-connecting hub, which is bolted onto the gearhead. Four press-fitted pins transfer torque from the hub to the wheel, and a custom shaft securely clamps the wheel between the hub and itself with three bolts. The use of three bolts instead of one adds strength to the design; a single bolt could easily become detached, but since the load is symmetrically distributed between three bolts, this is less likely.

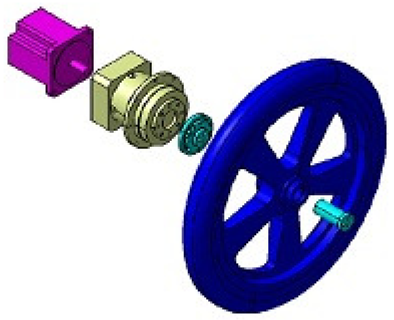


Figure 2: Drive Train

5. ELECTRICAL & ELECTRONIC SYSTEMS

Based on previous performance, upgrades and enhancements were added to improve safety, efficiency, and functionality.

5.1 Power Distribution

To provide adequate power to all sub-systems of *Bazinga!*, the requirements of each were first assessed under normal and worst-case conditions. The power for *Bazinga!* comes from four 12 V, 22 Ah Amstron sealed lead acid batteries connected in series, for an output of 48 V and is distributed via a custom-designed PCB to the vehicle sensors, wireless router, motors, and two Macbook Pro computers. The overall power distribution scheme is shown in Figure 3.

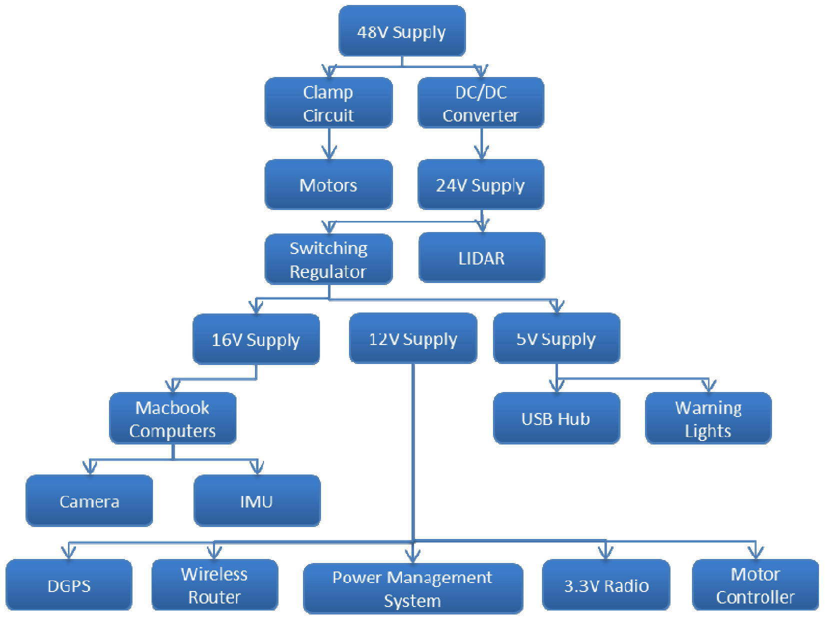


Figure 3: Power Distribution Scheme

5.2 Electronic System Protection and Safety

To help ensure that *Bazinga!* is safe, reliable, durable, and easily serviceable, several special features have been incorporated into the power distribution system. The PCB is designed such that high power components are isolated from lower power components. Fuses are strategically positioned on the PCB to prevent electrical damage due to unexpected current surges. The incorporation of high efficiency switching regulators provides stable outputs with low ripple. In addition, these regulators have been designed to protect the PCB from low battery voltage levels, short circuiting, and overheating, thereby extending the life cycle of the circuitry. A clamper circuit is connected to the motor power supply to absorb the motor's back EMF. The status of the power box is conveyed via a series of panel-mounted light emitting diodes (LEDs). Finally, vehicle-wide systems integration is addressed by the use of a real-time current and voltage monitoring system that sends status information from the power box to the main laptop through a USB connection. Thus, if a problem occurs, its source can be located quickly and diagnosed.

Beyond protecting *Bazinga!* from possible internal problems, the electronics are protected from both water and dust. The vehicle is weather proofed such that light rain will not cause electrical short circuits. This involves the incorporation of NEMA enclosures for the power distribution system, as well as a shell that surrounds the vehicle chassis and the various components.

Lastly, for user safety, *Bazinga!* is equipped with hard, soft and remote E-stops controlled by a mechanical button, the microcontroller, and the remote control respectively. The remote control, which can operate in one of two modes, Computer Controlled (PC) or Remote Controlled (RC), is made up of a custom designed PCB housed within a durable Futaba remote control shell. When the remote control is set to operate in PC mode, it transfers control of the motors to the computer (retaining E-stop control). If placed in RC mode, the operator can manually drive the vehicle.

The transceivers that are used in *Bazinga!'s* design are Aerocomm AC4490-200A transceivers. Although the vehicle only needs to be controlled from a maximum distance of 50ft, with the implementation of the aforementioned transceivers and full antenna extension, the vehicle is capable of being controlled from nearly a mile away. A twist-to-release remote E-Stop button is integrated into the remote control unit. As an added measure of security and immunity to interference, we transmit encrypted data over a spread spectrum wireless link for two-way communication between the vehicle and remote.

5.3 Sensor System

Bazinga! incorporates five sensors into its compact design: a camera, a LIDAR, a DGPS, a digital compass, and an IMU. Each sensor is enclosed in a waterproof case and firmly mounted to the vehicle while, at the same time, permitting relatively easy removal for servicing. The following is a brief description of the sensors that are used by *Bazinga!* as shown in Figure 4.

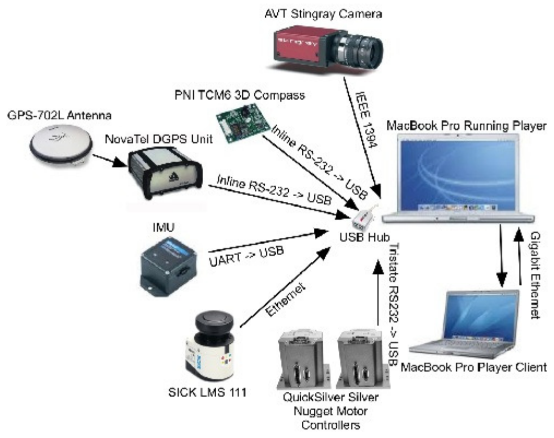


Figure 4: System Communication

Camera: The AVT Stingray F-080C 1/3" CCD camera was selected as the vision sensor for this vehicle. This camera uses the IIDC IEEE 1394B protocol to relay images, which is ideal for machine vision applications, because the frames are uncompressed and various options such as region of interest and lookup tables can be set and executed in hardware. Also, the camera's progressive scanning and high frame rates minimize motion blurring. The CS-Mount lens design enables the camera to accept a very wide-angle low-distortion lens, which provides a 125^0 field-of-view, which makes navigation heuristics easier to implement.

LIDAR: A 270^0 SICK LMS111 LIDAR unit was employed for the purposes of obstacle detection. The unit is capable of collecting data over a 270° field-of-view with 0.25° resolution, a maximum range of 20 m, and a 25 Hz scanning rate.

DGPS: To obtain positioning data in the Navigation Challenge, Novatel's ProPak-LB Plus DGPS system was selected. The DGPS antenna is mounted to the top of the vehicle's mast while the receiver is securely positioned inside the chassis. Using Omnistar HP's DGPS system, the signal is corrected to provide up to +0.1m accuracy. This system provides data at a rate of 20 Hz, which is adequate for *Bazinga!'s* expected speed and desired performance.

Digital compass: Sparton GEDC-6 was integrated into the vehicle to help determine vehicle heading. This compass provides a heading accuracy of 0.5° and updates at 20 Hz, which again is sufficient for the vehicle's speed and desired performance.

IMU: MicroStrain's 3DM-GX2 is a high-performance gyro enhanced orientation sensor which utilizes miniature MEMS sensor technology. It contains a 3-axis accelerometer, 3-axis gyro, 3-axis magnetometer, temperature sensors, and an on-board processor running a sophisticated sensor fusion algorithm. The 3DM-GX2 outputs include Euler angles, rotation matrix, deltaAngle & deltaVelocity, acceleration and angular rate vectors with a maximum data refresh rate up to 256 Hz.

5.4 Data Communication Interfaces

The various electrical and electronics systems were interfaced in the manner illustrated in Figure 4. The AVT Stingray camera is connected via Firewire (1394B) to the MacBook Pro. The DGPS system is connected to the computer through a USB interface using an inline RS-232-to-USB adapter. The SICK LMS111 LIDAR is connected to the computer via an Ethernet link. The Sparton GEDC-6 compass also uses RS-232. Finally, all the computers are networked via gigabit Ethernet.

6. SOFTWARE DESIGN

The primary goal concerning software this year was to improve upon the last competition entry's software structure, integration, and performance.

6.1 Development Environment

This year the *Bazinga!* team continued the use of a multi-layer environment for software development shown in Figure 5. These layers make communication between hardware and software possible. The first is the server layer which is occupied by Player™, an open-source, Unix-based (Linux or Mac OS X) robotic software system that serves as an interface between robotic algorithm implementations and the vehicle systems. Specifically, it provides standard interfaces for a typical set of robotic peripherals (LIDAR, cameras, motors, etc.) that can be accessed from the local computer or any other computer over a TCP/IP network. The second layer is the client layer occupied by user programs written in MATLAB. These layers communicate with Player™ over a TCP socket to acquire data from sensors and send actuator commands, which Player™ then passes on to the vehicle. The writing of efficient wrappers to enable communication between the two layers is what has enabled our program developers to utilize MATLAB®, with its wealth of proven and powerful resources, for algorithm development.

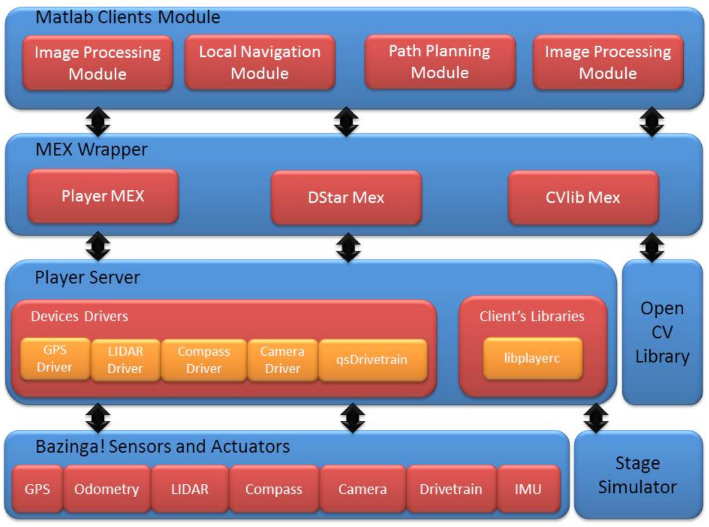


Figure 5: Software Development Architecture

6.2 Data Acquisition and Processing

With the need to process data from numerous sensors in a timely manner, an architecture that supports the use of multiple computers is used.

Sophisticated vision algorithms are central to a successful strategy for the Auto-Nav Challenge due to the complex obstacle, lane, and other terrain features present. If, when implementing these algorithms, the associated computational complexity causes the overall image-frame processing rate to drop too much, the vehicle may not be capable of operating effectively at higher speeds.

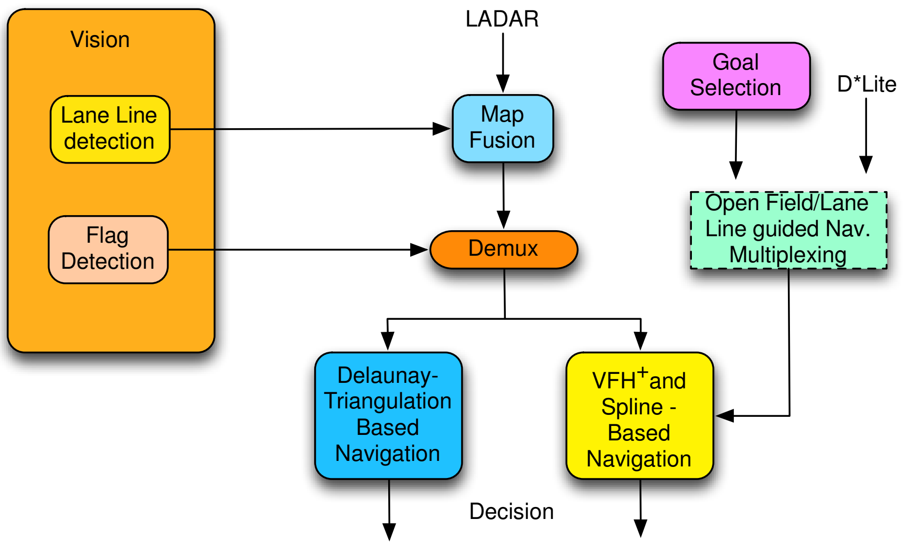
In order to favorably address this tradeoff, a multi-pronged strategy to increase the frame rate was adopted. *Bazinga!* distributes its computational tasks over two laptop computers (with a total of 4 processor cores). Using Player™ as a server facilitates setting up this distributed computing architecture and provides a wealth of existing open-source code to draw from. The top computer, a MacBook Pro, is entirely devoted to running components of the overall vision algorithm. This computer utilizes the MATLAB® parallel processing toolbox to spawn multiple workers which exploit parallelism in several of the IP routines (e.g., color segmentation and adaptive thresholding). This enables effective utilization of the multiple processor cores.

6.3 Software Design: Auto-Nav Challenge

The software architecture for the Auto-Nav Challenge consists of three blocks – a lane-guided navigation block, an open-field navigation block, and a mode-switching block between these two cases. The algorithmic flow is shown in Figures 6–8.

6.3.1 Lane-Guided Navigation:

This consists of six main blocks – vision, map fusion, Delaunay-triangulation-based navigation, VFH+ and spline-based navigation, goal selection and a mode switch (see Figure 6).



Vision
Lane Line detection
Flag Detection
LADAR
Map Fusion
Demux
Goal Selection
D*Lite
Open Field/Lane Line guided Nav. Multiplexing
Delaunay-Triangulation Based Navigation
VFH⁺and Spline - Based Navigation
Decision

Figure 6: Lane Guided Navigation Software Diagram

- Vision – Performs the tasks of identifying lane lines and flags.
- Map fusion – Combines LADAR obstacle information with lane line information.
- Delaunay-triangulation-based navigation – Is used to perform the flag-guided navigation.

- VFH+ and spline-based navigation – Is a local navigation algorithm that enables optimized motion planning.
- Goal selection – Keeps track of the forward direction and prevents vehicle turn-around.
- Mode switch – Determines whether to use Delaunay-triangulation-based or VFH+/spline-based navigation.

6.3.2 Open-Field Navigation:

This consists of three blocks – Global mapping, D*Lite and VFH+/spline-based navigation (see Figure 7).

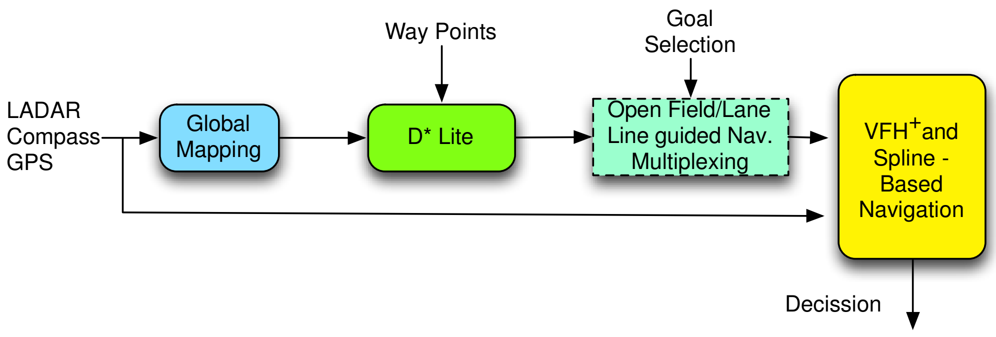

Way Points
Goal Selection
LADAR
Compass
GPS
Global Mapping
D* Lite
Open Field/Lane Line guided Nav. Multiplexing
VFH+and Spline - Based Navigation
Decission

Figure 7: Open-Field Navigation Software Diagram

- Global Mapping – This builds a progressive global map as the vehicle navigates the course.
- D*Lite – Generates breadcrumbs leading to the next waypoint.
- VFH+ and spline-based navigation – A local navigation algorithm that enables optimal motion planning.

6.3.3 Mode Switch:

This determines which one of the above modes should be used to navigate the robot. The given waypoints are used to form a convex hull. Anytime the robot is inside the hull, open-field navigation mode is activated. Otherwise lane-guided navigation is used (see Figure 8).

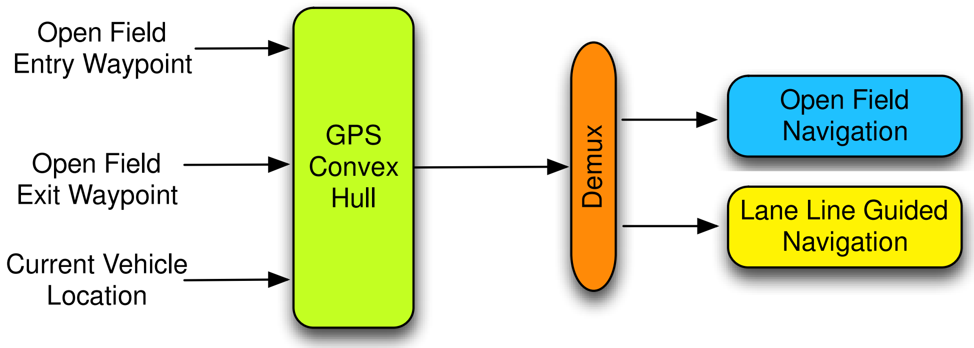


Figure 8: Mode Switching Logic

6.4 Legacy Algorithms:

Four legacy algorithms are ported from Cerberus into Bazinga!, The lane line detection, goal selection, global mapping, and path planning algorithms.

6.4.1 Lane Line Detection - Vision

To address the increasing complexity of the vision task in the Auto-Nav Challenge course, the *Bazinga!* team began by adopting the proven solution from Cerberus's vision methodology. In the past, images were captured in RGB format, which represents the image color components as red (R), green (G), and blue (B). Lane lines were then identified based on a gray scale image that promotes the lane line color combinations and demotes those of other objects in the scene. This grayscale image was derived with simple scaled additions of the R, G, and B planes. Recently, colored barrels have been added to the autonomous course, and have made the gray-scale image formula and thresholding very sensitive to lighting conditions and scene content. Therefore, *Bazinga!* has moved to a heavily color-dependent recognition algorithm that is more stable and robust.

HSV Thresholding: *Bazinga!* processes images using hue (H), saturation (S), and value (V) planes. One aspect that makes the HSV representation appealing is that the lighting information is contained in the value plane, while the color information can be found using the hue and saturation planes. Each pixel in the image can be defined by these three values. The hue specifies the color, the saturation specifies the amount of color, and value specifies the intensity information. The image processing algorithm identifies lanes and obstacles. This is particularly useful since the LIDAR is unable to identify lanes as well as asymmetrical saw horses. The output of the image processing algorithm is passed on to a data fusion process that fuses the LIDAR readings and lane-obstacle information into one obstacle data set.

The image processing algorithm works as follows: first, the captured RGB image is converted into an HSV image, and then thresholding is applied to the hue and saturation planes to remove grass from the image. This leaves behind anything that is an obstacle or is a lane. Finally, thresholding is performed again to extract anything that is white. In the HSV representation of an image, anything that is white will have a fairly low saturation value and can have almost any hue or intensity value. The algorithm then checks for connected components and removes anything that is small and thus considered to be noise. Figure 9 shows a captured image and the corresponding output of the vision algorithm.

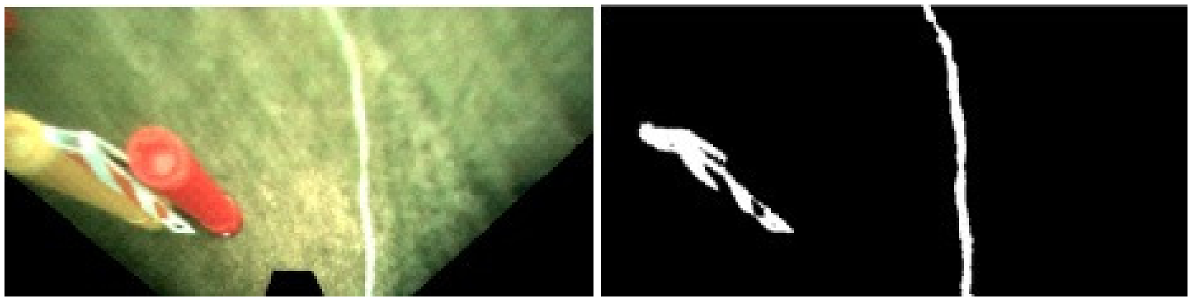


Figure 9: Captured Image on Left/Processed Results on Right

6.4.2 Goal Selection

The goal selection algorithm is concerned with determining the "forward" direction. For the Navigation portion of the Auto-Nav Challenge this is relatively easy, as forward is towards the next waypoint. However, in the Autonomous portion of the Auto-Nav Challenge, the forward direction is less easy to determine, since it requires the vehicle to "go around the course" without turning around. The forward direction has to be established from the results of the vision algorithms, which are not 100% reliable. Further complicating the situation is the presence of course features such as switchbacks and ramps, which can create apparent traps.

To deal with this situation, a heuristic layer is created. This layer combines two pieces of information to set the goal direction. The first is a "GPS history" direction established by GPS coordinates 4 meters apart obtained from the immediate travel history of the robot. The second is the result of the vision algorithm, which has a built-in level of confidence measure. When the vision results are less reliable, greater reliance is placed on the tail in setting the goal direction and vice versa. This approach contributes to improved navigation of switchbacks and traps.

6.4.3 Global Mapping:

Mapping is an important capability for autonomous robots that facilitates good decision-making. It could be beneficial in the Nav Challenge part of the Auto-Nav Challenge, if the robot has to return to territory through which it has passed, since it enables the generation of an optimal route between waypoints. However, with the new Auto-Nav competition formation, this is less likely to be useful compared to previous years. In the Autonomous portion of the Auto-Nav Challenge, the ability to maintain a global map could be used to overcome momentary lapses in the accuracy of the vision algorithms, while also providing an ability to carry accumulated terrain knowledge into the next heat. Mapping requires an accurate estimate of the robot pose (X, Y, Yaw) so that precise registration of the local map on the global map can be carried out. A Kalman Filter was implemented to estimate vehicle pose reliably by fusing data from the motor encoders, the DGPS, and the digital compass. This also allows GPS outages to be managed if they happen to occur.

6.4.4 D*Lite Path Planning:

Performance in the Navigation portion of the Auto-Nav Challenge is considerably enhanced if path planning is utilized. Path planning is carried out using the D*Lite algorithm, which provides the best route between waypoints and avoids getting trapped when faced with a concave obstacle configuration. D*Lite can work off a partially complete map of the field, and progressively re-plans the optimal route when the map is augmented with new information as the robot explores.

7. INNOVATIONS

7.1 Software Innovations:

7.1.1 VFH$^+$ and spline – based navigation:

The previous UDM implementation of VFH$^+$ did not take into account vehicle dynamics. The robot was assumed to have the ability to move in any direction instantaneously, regardless of its current heading or linear and angular velocities. In reality, when the robot decides to change directions, its trajectory is defined by specific curves, which are dependent on the current linear and angular velocities. Figure 10, shows the different robot trajectories without dynamics (a) and with dynamics (b). Taking into account the dynamics of the robot allows it to avoid obstacles more efficiently. In the updated implementation of VFH+, combined obstacle and curvature polar histogram is generated. This histogram allows the robot to choose only navigable sectors and improves the path contouring.

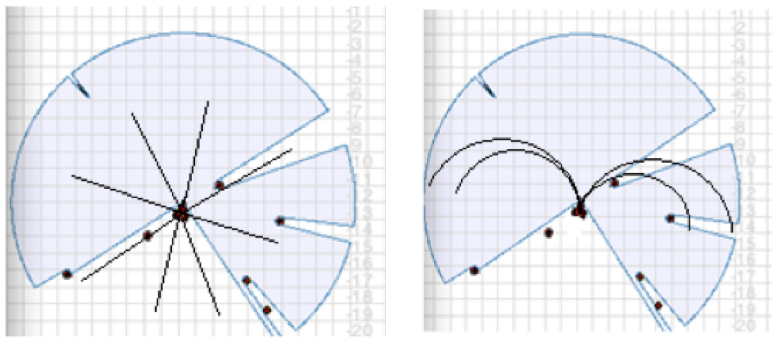


Figure 10: a) without dynamics b) with dynamics

In the past, our vehicle maneuvered slowly. This year, we aim to maximize the speed at which our robot moves by using speed modulation. The first part of this Effort lies in getting the robot to follow smooth curves rather than "point & shoot" navigation. Following smooth curves allows the robot to take advantage of its momentum while maneuvering. These smooth curves are generated using parametric equations based on the robot's initial velocity and a chosen goal point.

Since we are now traveling along curves rather than according to a chosen heading, preventing collisions becomes more complex. The key to preventing collisions despite following curved paths is an appropriate choice for the target point and final bearing of the curve. By looking at the distance of the obstacles on either side of the chosen goal direction, w select an appropriate target point that ensures the robo not hit any obstacles on its way to the target sector. E of smart endpoints are shown in Figure 11. As a final precaution, we have the ability to plot our chosen route to the target against our map of known obstacles.

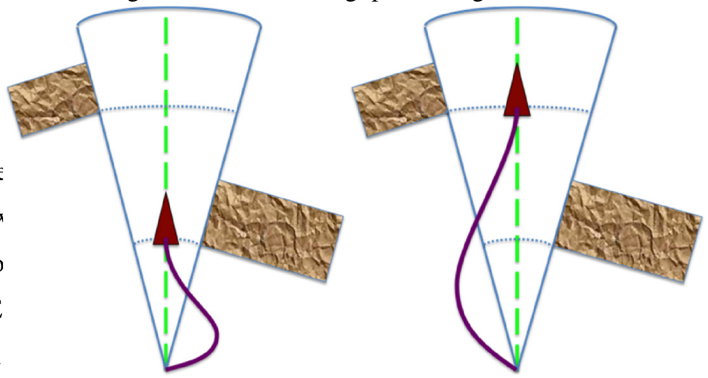


Figure 11: Preventing Collisions with Smart Endpoints

The final part of speed maximization involves the physical capabilities of the robot. The first consideration involves choosing the final velocity when we reach the target point. We choose our final speed along the path based on the braking ability of the robot and the distance from the

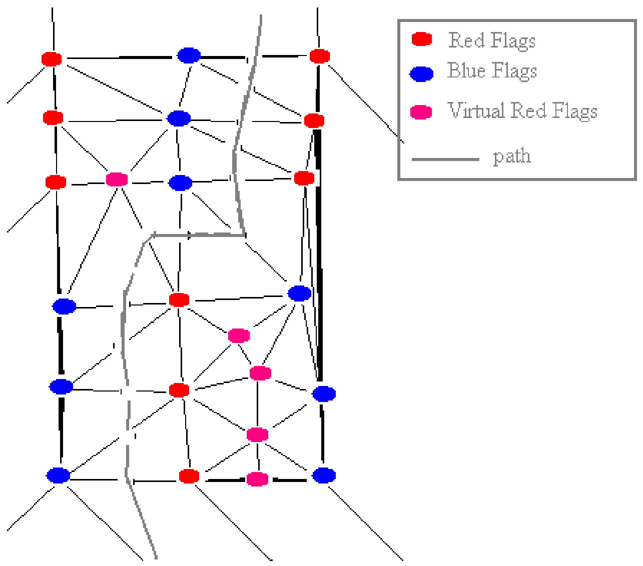
target point to the nearest obstacle or unexplored territory. We want to still have the ability to stop if as we go forward, we suddenly notice a brick wall. The second consideration is the speeds and accelerations. For any proposed path, we calculate the tangential acceleration, normal (centrifugal) acceleration, overall acceleration, turn rate, angular acceleration, vehicle speed, and wheel speeds & accelerations, and we choose the quickest path that is within the limits imposed by all these physical quantities.

7.1.2 Flag detection algorithm:

New to the autonomous challenge are the red and blue flags. Once identified, the flags will be used to guide the robot in a specific manner. To accomplish this task, color segmentation using Hue, Saturation, and Value (HSV) was used. By knowing the specific hue of the flags over a threshold saturation, the flags can be identified. The hue will relate to the color of the flag and the saturation will relate to how much of the color is present. As the saturation increases, the color becomes less white and closer to the color of interest. Segmenting the images independently for each color will result in a binary image that can be used to describe the locations of the flags present in the image. These locations can then be used in conjunction with an enhanced navigation strategy to maneuver between the colored flags.

7.1.3 Delaunay-triangulation-based Navigation

In the flag lane section of the course, a different approach based on the concept of Delaunay Triangulation is adopted for robot navigation. The robot is required to stay to the left of the red flags and to the right of the blue flags. As explained above, the vision algorithms are designed to detect the location of the colored flags. Middle points are generated for the Voronoi edges and used as breadcrumbs for navigation using VFH. Additionally, virtual red flags are placed to the right of detected red flags to block passage (see Figure 12). An elaborate heuristic is developed to deal with multiple detection cases.



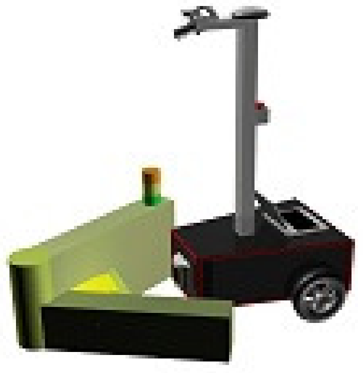


Figure 13: Docking Station

Figure 12: Delaunay Triangulation for Flag-guided Navigation

7.2.1 Charging Station:

7.2 Hardware Innovations:

A significant hardware innovation for *Bazinga!* is the self-charging docking station for the robot (see Figure 14). The power box sends a low power signal to the computer, upon which it initiates the docking/charging procedure. The robot navigates to the proximity of the docking station using its known GPS location. It then executes a slow 360º turn-in-place maneuver to find the opening of the station and then enters the opening making the necessary fine adjustments through a PID control algorithm. Contact with the specially-designed connector arrangement is made and the charging takes place under the supervision of the microcontroller.

8. SYSTEM INTEGRATION

This project was divided into subtasks to facilitate development and assignment of tasks to individuals. However, this then requires a process to integrate all the parts into a single, working product. The team utilized the Player/Stage™ platform from the beginning. All hardware interaction was done through Player's common interface. This meant that all algorithmic code, being a Player client, could be developed and tested using the Stage™ driver set. Software system integration for *Bazinga!* was tested on a test course built on campus to represent possible scenarios that might be encountered at the IGVC. The physical integration of the subsystems, to fit properly and make efficient use of space, was given careful consideration during the chassis design process, by taking their dimensions and locations into account. This allowed the designers to allow for the expected footprint of each subsystem as well as the space needed for their interconnection.

9. JAUS

Our goal for the JAUS challenge was to implement a system that would meet the SAE JAUS requirements, while at the same time be compatible with each of the research robots used in our Advanced Mobility Lab (see Figure 15). We chose to implement JAUS using Jr Middleware. It is SAE AS-4 AS5669A compliant software that handles routing JAUS messages on a network. It provides an API that allows a program to create and send out a message through the use of a function call.

Our JAUS implementation interfaces with Player to obtain information, and perform JAUS-related tasks. The advantage is that our code could be compiled and run on any system that uses Player. To verify the functionalities of the system a COP program was written that would send out the messages expected at the competition and display the incoming messages from our system. This allowed us to verify the functionality of messages that are not supported this year by the JAUS validation tool.

10. PERFORMANCE

10.1 Speed

The theoretical maximum speed of *Bazinga!* is approximately 5.67 meters/second given its 40.6 cm wheel, 12:1 gear ratio, and 1600 rpm speed at optimal torque. However, the maximum speed that *Bazinga!* will reach during competition is capped at 2.24 meters/second (5 mph).

10.2 Ramp Climbing Ability

Based upon the rated torque output of the motors, the size of the vehicle's wheels and the selected gearing, calculations and testing have revealed that *Bazinga!* has ample torque to ascend an incline with a gradient of up to 30% (16.7°) without stalling. According to the IGVC rules, the vehicle needs only to be capable of climbing a 15% (8.5^0) incline.

10.3 Reaction Time

For the Autonomous Challenge, it takes approximately 100 ms (10 frames per second) to run the system algorithms (based on software timing estimates). At 5 mph, which is the maximum speed for *Bazinga!*, this cycle time translates to a decision being made approximately every 22 cm of travel. In the Navigation Challenge, the algorithms take approximately 40 ms to complete. At the 5 mph speed limit, this cycle time corresponds to a decision being made approximately every 9 cm.

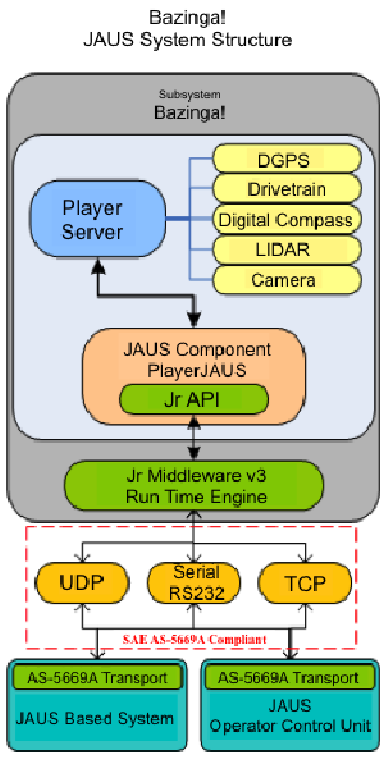


Figure 14: Bazinga! JAUS Architecture

Table 2: Power Consumption Estimates

Power Distribution						
	Normal Conditions			Worst-Case Conditions		
Device	Volts	Amps	Watts	Volts	Amps	Watts
LIDAR	24	0.4	9.6	24	0.5	12
Laptop	16.5	2	33	16.5	3.6	59.4
Laptop	16.5	2	33	16.5	3.6	59.4
DGPS	12	0.2	2.4	12	0.2	2.4
Compass	5	0.02	0.1	5	0.02	0.1
Camera	12	0.17	2.04	12	0.17	2.04
Motor/Controllers	48	3	144	48	10	480
Wireless Router	12	0.5	6	12	0.5	6
TOTAL POWER			**230**			**621**

10.4 Battery Life:

Table 2 lists the power consumed by the vehicle components under normal as well as worst-case operating conditions. Using these values, it is expected that the vehicle will be able to run for approximately 4.5 hours under normal operating conditions and slightly less than 2

hours under the worst-case conditions. These estimates have been exceeded in actual runs. Two 288 W DC battery chargers are positioned inside the vehicle to enable the batteries to be recharged.

10.5 Obstacle Detection Distance

The LIDAR unit on the vehicle is capable of detecting objects at a distance of 20 meters; for *Bazinga!*, the LIDAR is configured for a range of 10 meters. The camera is set up to view a shorter range to reduce glare and horizon effects (approximately 5 meters).

10.6 Accuracy of Arrival at Waypoints

The waypoints at the competition will be designed as concentric 2m and 1m radius circles centered on the GPS coordinates of the waypoints. *Bazinga!'s* DGPS system provides an accuracy of + 0.1 meters in DGPS mode, and + 0.01 meters in real-time kinematic (RTK) mode. It can be seen that this accuracy is more than sufficient. This has also been demonstrated both via simulation and actual experimentation.

11. SAFETY, RELIABILITY, DURABILITY

Even though *Bazinga!* is a developmental vehicle, it is important for it to operate in a safe and reliable manner as well as be durable, just like any other product. The durability of its mechanical and electrical/electronic systems can be counted on because the design builds upon what worked well in our previous vehicles, while at the same time upgrading and/or redesigning what needed to be improved. *Bazinga!* includes several features that not only contribute to its performance, but also increase its safety, reliability, and durability. Three E-Stop systems are implemented to ensure that the vehicle can be stopped safely, quickly, and reliably. These are the soft, hard, and remote E-Stops, which are controlled by the microcontroller, the manual mechanical button on the rear of the vehicle, and the remote control, respectively. The vehicle is weatherproofed such that light rain will not cause electrical short circuits. This involves the incorporation of NEMA enclosures for the power distribution system, as well as a shell that surrounds the vehicle chassis and the various components. All electrical circuits are carefully fused to prevent electrical damage. Furthermore, individual currents and voltages are monitored in all circuits. Diagnostic software and LED indicator systems were developed so faults could be quickly identified and repaired.

Bazinga! implements three levels of "watchdogs" on the motor controllers to prevent unintended vehicle operation. The first watchdog is a hardware watchdog, which prevents vehicle operation in the event of a hardware failure. Every 500 ms the computer must send a specific message to the motor controller. If the message is not sent, an E-Stop is triggered. In the event of a hardware failure or computer crash, the message will not be received by the controllers and the vehicle will stop. The second watchdog is a software watchdog, to prevent vehicle operation in the event of a software failure. The motor driver will expect a new velocity command from the software algorithm at least every 2 seconds. If such a command is not received, the driver will halt the motors until a new command is received. The third watchdog monitors smooth wireless data transmission between the remote control and the vehicle. Any failure of the remote control or jamming of the wireless signal will trigger the E-Stop, acting as a hardware watchdog.

12. CONCLUSION

The UDM team is excited at the prospect of attempting to reclaim the IGVC title after a gap of a year with *Bazinga!*. There have been significant changes in the competition rules since our previous entry and we have responded by incorporating the necessary changes in our vehicle design. We will use the remaining days before competition working on improving system integration and optimizing performance.