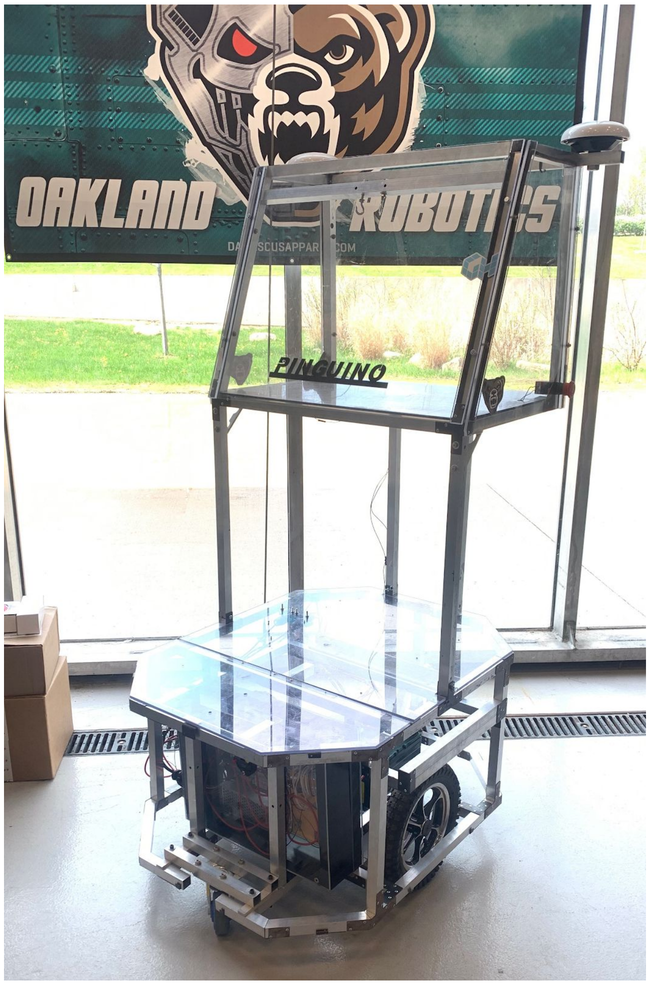
IGVC 2019 - PINGUINO

THE PINGUINO INTELLIGENT ROBOTIC PLATFORM



TEAM CAPTAIN:
Nicholas Musienko (nmusienko@oakland.edu)

TEAM MEMBERS:
McKenzie King (mckenzieking@oakland.edu)
Mackenzie Hill (mlhill@oakland.edu)
Trevor Stiteler (tjstiteler@oakland.edu)
Michael Lohrer (mflohrer@oakland.edu)
Austin Fee (austinfee@oakland.edu)
Brandon Kujawa (btkujawa@oakland.edu)
Andrew Dimmer (adimmer@oakland.edu)
Andreas Eickhoff(ateickhoff@oakland.edu)
Nolan Greene (nolangreene@oakland.edu)
Alyssa Musienko (amusienko@oakland.edu)
Ian Matthews (ianmatthews@oakland.edu)
Carolyn Jordan (csjordan@oakland.edu)
Robert Brown (rbrown3@oakland.edu)
Christopher Mackenzie (cmackenzie@oakland.edu)
Antonio Scalzi (ascalzi@oakland.edu)
Anthony Giannattasio (agiannattasio@oakland.edu)
Nicholas Gracey (nlgracey@oakland.edu)

STATEMENT OF INTEGRITY:

I certify that the design and engineering of Pinguino by the current listed student team has been significant and equivalent to what would be awarded credit in a senior design course at Oakland University.

[illegible]

Faculty Advisor: Dr KaC Cheok Ph.D.
Date Submitted: May 15th, 2019

DESIGN PROCESS AND TEAM ORGANIZATION

Introduction

Oakland University is proud to enter Pinguino into the 27th Annual Intelligent Ground Vehicle Competition (IGVC). Pinguino includes a two-wheeled differential drive platform with a sensor tower. Custom electronic systems providing motor control and sensor integration were designed and built to meet the specific requirements of the IGVC vehicle. All software systems, including vision processing and map-based path planning, were simulated and integrated in the Robot Operating System (ROS) environment.

Organization

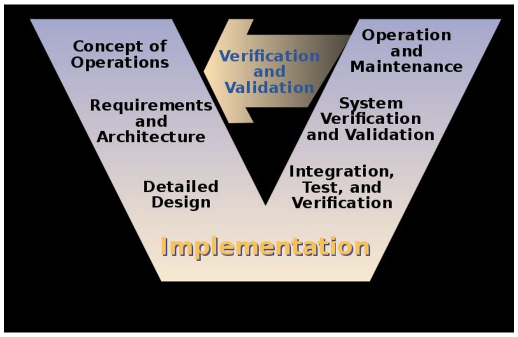
Due to the complexity of Pinguino's autonomous robotic platform, the overall team is broken into three subteams to insure the mechanical, electrical, and software requirements are met. These electrical, mechanical, and software subteams work in tandem to complete their individual assignments as the robot is built from the ground up. Each subteam is led by a Subteam Lead who reports to the Team Captain at the end of each meeting. Bi-weekly progress reports are prepared by the three Subteam Leads and presented to all team members to aid in integrating and maximizing the parallel work schedules. To accomodate members unable to attend the bi-weekly progress report meetings, the reports are posted online where they can be viewed and commented in order to maintain a living record of up-to-date information.

Design Assumptions

Based on the Official Competition Rules, constraints pertaining to the dimensions, safety, speed, and payload of the vehicle were considered during the designing of Pinguino and each met respectively. Because the rules of the competition hardly vary from year to year, knowledge gained from previous competition years can be used in the design for the current year's entry. Due to this previous experience, Pinguino was created with a differential drive system in order to most efficiently avoid obstacles. Other aspects implemented in the platform include the frame and basic body design, inspired by the Oakland University team's IGVC entry back in 2017.

Design Process

To build off the initial design assumptions, a classic "V-Model" design process was followed to further develop Pinguino. With established and defined requirements, a design was formed using CAD software. This new model was then tested in a team-created environment to develop the navigation system. After implementing the design and integrating the various components, a rigorous test cycle began in which consistent failure points were identified and rectified through minor adjustments or larger design changes.


Concept of Operations
Requirements and Architecture
Detailed Design
Verification and Validation
Operation and Maintenance
System Verification and Validation
Integration, Test, and Verification
Implementation

EFFECTIVE INNOVATIONS IN VEHICLE DESIGN

Innovative Concepts From Other Vehicles Applied to Vehicle

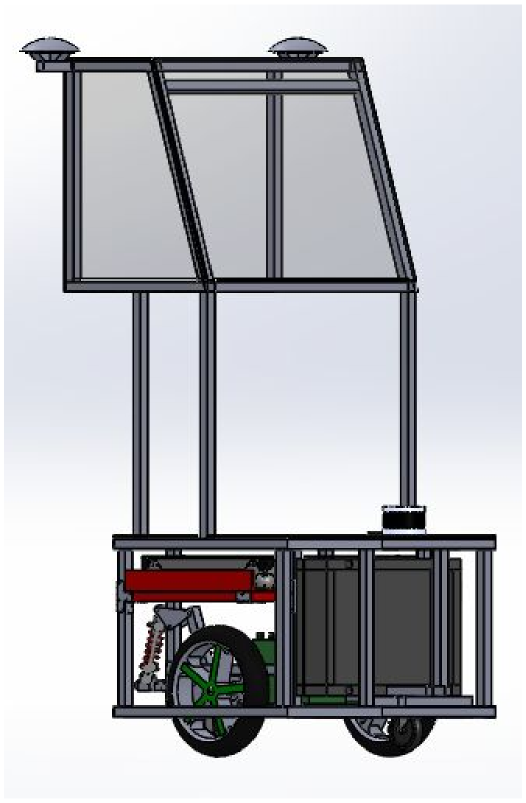
The previous platform, Mantis, provided the team with the inspiration to ensure the robot could be programed while out in the field testing. Team members can program any aspect of Pinguino from the laptop located in its body, at a comfortable position for use without unmounting it from the robot. From this platform, we also modeled our electronics boxes. Mantis was the first robot from Oakland University to feature a electronic box to house the electronics, and Pinguino varies that formula by adding a second electronics box. One last thing implemented from Mantis is a dual GPS antenna setup. This placement provides the heading of the robot through not only long term GPS data, but from single data points.

Innovative Technology Applied to Vehicle

One innovative concept employed in Pinguino is a Kangaroo Self-Turning motion controller. This provides a self-tuning PID feedback loop to ensure encoder data is correct and to automatically compensate for any deviant values. Another innovative technology used was IP67 waterproof keyed connectors for all of the electronics boxes. The waterproofing keeps the electronics as safe as possible from the elements seeing as the competition is outdoors in virtually all weather conditions. The keying in the connectors is used to ensure our connectors are all visually similar and use the same parts while making it impossible for them to be plugged into the wrong place or in the wrong fashion.

DESCRIPTION OF MECHANICAL DESIGN

Overview



The mechanical subteam is tasked with designing and manufacturing a frame and all mechanically related accessory components required for the physical vehicle. Additionally, the mechanical team is responsible for solving any design related problems including vehicle functionality, manufacturing feasibility, design constraints, and user-vehicle interaction. In order to comprehensively determine the constraints of the design, the mechanical team works closely with the electrical and software teams to ensure that the final vehicle design accommodates the needs of each subteam. The aluminum frame consists of a base and tower while accessory components include electronic boxes, motor/battery mounts, custom suspension, weatherproof covering, and sensor mounts. Pinguino features an octagon base that houses all electrical components and a tower used for a majority of sensor placement. All mechanical aspects of Pinguino adhere to the vehicle configuration and qualification requirements as laid out by the official competition details provided by the 27th Annual IGVC.

Frame

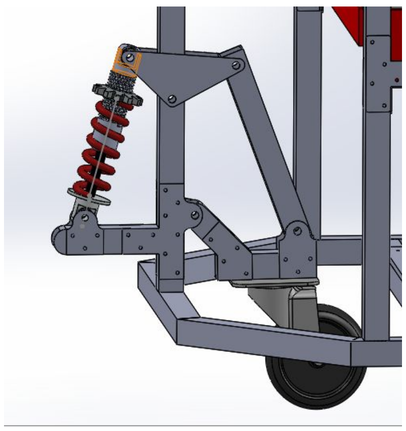
The custom built frame is made from hollow aluminum tubes connected via steel brackets. These brackets are custom designed for different joint placements and manufactured in-house using a water jet. Aluminum tubes are used to construct the frame to reduce the overall weight of the vehicle while maintaining structural integrity. Steel brackets were chosen since a stronger metal is ideal for better joint stability. Additionally, clear plexiglas is used to cover the top external surfaces of Pinguino to provide waterproofing for the internal components. The octagonal shape of the base is designed to reduce collisions between the vehicle's corners and the course obstacles. Pinguino also has a large body structure to house all internal components. Motor mounts were built on the center lateral sides of the base to attach the motor/wheel assembly. This wheel placement was chosen because it allows for a zero degree turning radius. Front and rear caster wheels are used to stabilize the base. The batteries are housed between the motors to assist in lowering the center of gravity.

Electronics Boxes

Two separate electronics boxes were built in order to keep the design modular and waterproof. One contains all components used to run and operate the controls components. The other contains components responsible for operating the various sensors. This allows the robot to run just the controls portion for teleoperational functionality as well as running just the sensors for software testing. The suspension is located center rear to provide cushioning when the vehicle traverses difficult terrain. The tower of the robot elevates and houses the cameras and GPS antennas to achieve their optimal positions for line perception and GPS signal/heading, respectively. The LiDAR is mounted in the front center of the base to allow for a 180 degree spherical view of what's in front of the vehicle.

Suspension

We have implemented a custom built rear suspension to improve robot handling. This suspension is linkage driven with a single pivot point. This design consists of a single strut, aluminum/bracket connectors, and a swiveling caster wheel to increase efficiency and decrease complexity compared to the old model. This year's design also features increased mobility and reduced internal space consumption from the suspension system. A spacer was also added to the strut to adjust preload and vary the stiffness of the springs to accommodate specific



terrain and/or weight of the vehicle. The softer spring allows for a smoother ride.

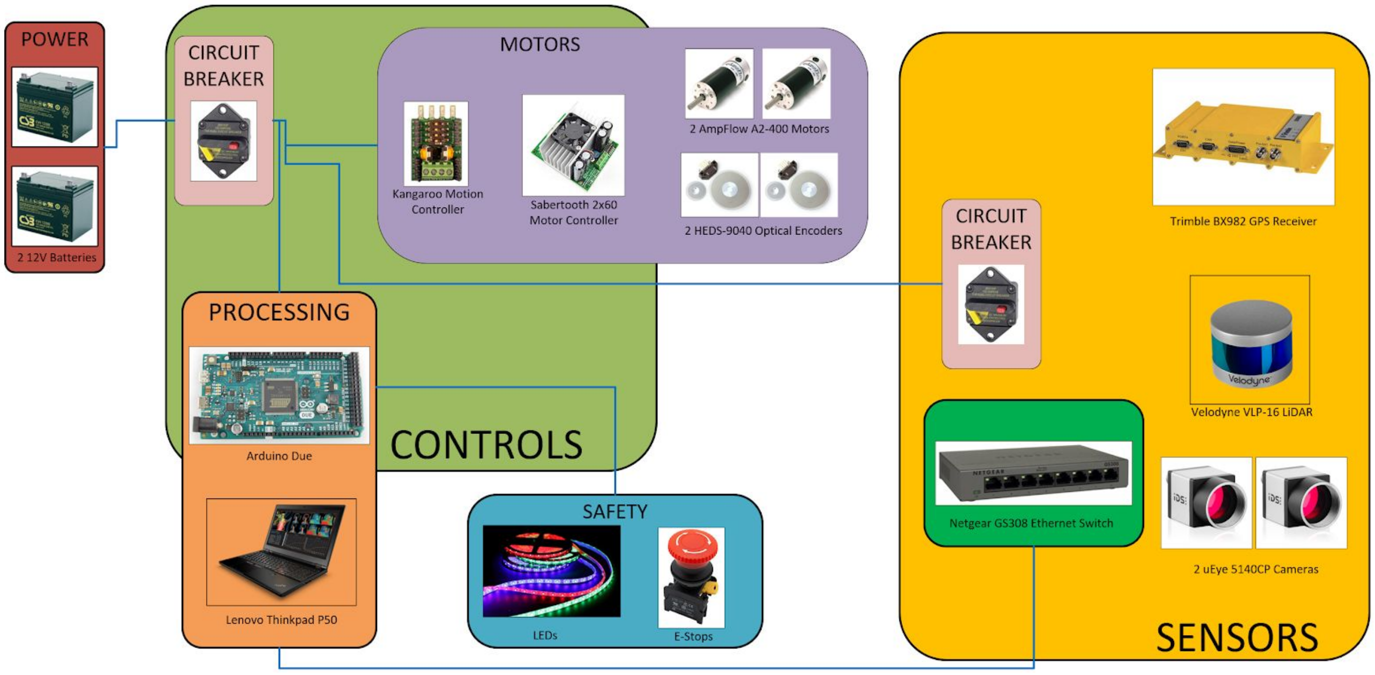
Weather-proofing

All of the connectors on the electronics boxes are IP67 rated when mated or capped. Plexiglas external covering on the frame protect internal components from hazardous weather and mud.

ELECTRONIC AND POWER DESIGN

Overview

The electrical system powering Pinguino is a completely new design built specifically for the platform's operation. To maintain overall system modularity, the majority of critical electrical components are enclosed within two electrical boxes (e-boxes). The control e-box contains an Arduino Due microcontroller, Sabertooth 2x60 motor controller, Kangaroo motion controller, fuse box, RC receiver, Inertial Measurement Unit (IMU), an isolated DC-DC buck converter, a custom perfboard containing critical circuits, and a circuit breaker. The sensor e-box houses the GPS, Ethernet switch, Power over Ethernet (PoE) injector, LiDAR breakout box, DC-DC boost regulator, DC-DC buck regulator, fuse box, and circuit breaker. By collecting all of the electrical components in two central locations, robot maintenance and modifications are made easier and the shortened length of data cables helps to prevent errors in data transmission. Each decision in the creation of Pinguino's electronic system was made with safety, reliability, efficiency, and modularity in mind. Several safety features have been implemented in hardware and software to ensure that users and bystanders are as safe as possible, as well as to mitigate any potential vehicle failure modes that might arise.



Power Distribution System

Pinguino is equipped with a system capable of supplying a constant 150A at 24V to the electronics. The power source consists of two 12V lead-acid batteries connected in series to achieve the desired 24V. Various sensors require 12V, and to do this two DC-DC buck converters are used to provide power within each e-box. Powering the cameras involves Power over Ethernet, in which the PoE injector requires 48V to operate. Generating this 48V requires a DC-DC boost converter to step the voltage up from 24V. The entry point for each electrical box is a circuit breaker rated for the load of the box, 150A to the control e-box and 60A to the sensor electrical box. Within each electrical box, a fuse box exists to ensure components do not get damaged, with separate critical parts getting separate appropriate fuses. Because electrical components exist within normally sealed electrical boxes, connectors are necessary to allow power to transfer across the boundaries of the boxes. The chosen connectors are rated to allow the appropriate power and are rated at IP67.

Electronics Suite

As with the twin electrical box setup, the electronics can be divided up into two major groups, controls and sensors. The control system was designed by Oakland University students around the Arduino Due microcontroller and a Sabertooth 2x60 motor controller. To ensure that the motors spin correctly, optical quadrature encoders are mounted to the shaft of each motor, their outputs getting routed to a Kangaroo motion controller. The Kangaroo is a self-tuning closed-loop PID controller designed specifically for communication with the Sabertooth, which allows for position and speed readings from the motors. Powered off the Arduino Due is a BNO055 IMU, for more precise measurements of movement. An RC receiver is also powered off the Arduino Due, and the RC transmitter acts as both a user input for the robot and as the emergency stop. Also included is a perfboard containing several custom circuits. An LED driver, battery voltage sensing circuit, logic step-down circuit, and software e-stop implementation are all custom-made for this application and are found on this perfboard.

The sensor system was designed via collaboration between the electrical and software subteams. An Ethernet switch collects the data from the GPS, LiDAR, and cameras for transmission to the computer.

Computer. The computer acting as the core of the robot is a Lenovo Thinkpad P50. This specific model boasts a 2.8-3.7GHz Intel Xeon Quad Core Processor, 16GB of RAM, a 256GB SSD, and an NVIDIA Quadro M2000M GPU. This laptop has the processing power necessary to collect the present data and run the appropriate computations.

GPS. The chosen GPS for the platform is the Trimble BX982 GPS receiver. It operates at 50Hz, and has an accuracy of up to a 8mm. For this GPS, there are two antennas, allowing for more accurate readings.

LiDAR. The Velodyne VLP16 LiDAR was chosen for its 16 scanlines and its accuracy of 3cm. It has an angular resolution of 2°, with a 360° field of view and a rotation rate of between 5 and 20Hz. It publishes its data over Ethernet and is capable of generating up to 0.3 million points per second. These specifications are plenty powerful for the given application, and provides enough data to determine an appropriate behavior.

Camera. Two uEye 5140CP color cameras are used in this platform. Each camera is capable of PoE operation, allowing both power and data to travel over a single Ethernet cable each. Each camera is capable of 88 frames per second operation with a 1280 by 1024 resolution.

Safety Devices and Integration

Many of the features found in Pinguino are there as safety measures. There are two physical emergency stop buttons mounted to the robot and a custom circuit within the controls e-box. If any of the three inputs are in the wrong state, relays connected to each output of the Sabertooth motor controller will disengage, disconnecting the motors from the control circuitry in hardware until the inputs enter the right states. The Arduino This stop mode is also togglable via software by a specific input from the RC remote, acting as a wireless E-stop that can be used across the 100 foot minimum requirement. Via the Sabertooth motor controller and the Kangaroo motion controller, the speed of the motors can be limited to a safe range, ensuring no violent behavior, and a firmware timeout can be set, turning the motors off if a command is not received within a certain amount of time. Lastly, there are plenty of fuses and circuit breakers to ensure safe battery discharge. All important devices have an inline fuse to prevent damage from a massive current spike, and the motors have in-line fuses to disconnect them in the case of excess current draw. Both the control e-box and the sensor e-box have circuit breakers integrated into their power distribution circuitry, allowing them to be turned on and off at will, and while also adding a last layer of protection by limiting the total power allowed to enter the system.

SOFTWARE STRATEGY AND MAPPING TECHNIQUES

Overview

Pinguino's software strategy uses the Robot operating system (ROS) as the framework and main method of integrating all of the software components. The system takes in sensor data from LiDAR, cameras, GPS and IMU. With the data from these sensors Pinguino is able to calculate the position of itself as well as the locations of nearby obstacles. The software employs sophisticated strategies to detect and locate obstacles such as lines and barrels. Using this information a map of our environment can be generated, the robot can be localized in that environment and plan a path to our goal.

Obstacle Detection and Avoidance

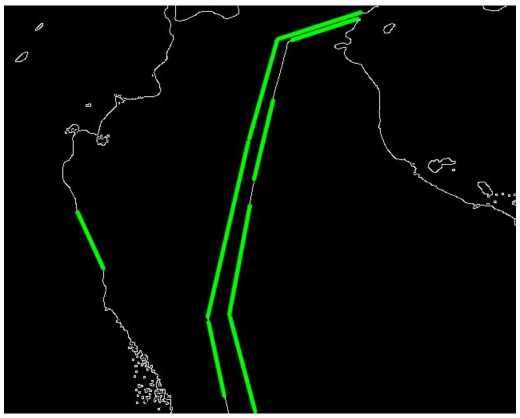
LiDAR. One of the sensors we use for our object detection and avoidance is a Velodyne VLP16. Alongside this, we use the Point Cloud Library (PCL) to process

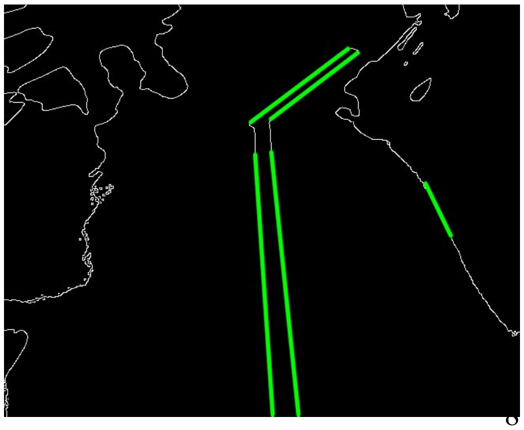
the resulting Point Cloud Data (PCD). We have implemented a primary vision node that run through a process to detect barrels in the way that follows:

1. Remove planes, the field, from the data.
2. Run through a cylindrical segmentation.
3. Advertise out the cylinders data in pcd format fitted to the frame.
4. Advertises out an array of every barrel's location and radius.

This program is also configurable to work with different parameters for distance filtering, removing unnecessary data, etc..

Cameras. In addition to LiDAR, cameras are also utilized in the software design. Pinguino includes two uEye cameras mounted at outward angles in order to view the lines at an optimal angle to facilitate the required line detecting and lane keeping needs. After the Probabilistic Hough Transform (this exports onto an image overlay), the next step was adding the lines to the cost map. To do so, the lines are converted into a point cloud that is relative to the base frame of the robot. Thus, instead of actually transferring the lines themselves onto the base frame and then the cost map, a semi-random selection of points that are collinear with the 2 endpoints are instead projected onto the base frame. So long as the points are close enough together that the robot thinks it cannot pass between them, a whole line does not need to be passed. In addition, by basing the line detection algorithm on the fact that the robot avoids the lines, as opposed to specifically following between them, the robot is better equipped to navigate stretches where the lines are far enough apart that the robot cannot see both lines on the track.





GPS. Absolute location information is provided by a duel antenna Trimble BX982 GPS. The GPS publishes position data at a maximum rate of 50Hz with an accuracy of up to 8mm. The duel antennas also provide the robot's heading down to 0.09°. This information is used in conjunction with the IMU and wheel speed data to accurately localize the robot on the course.

IMU. The Bosch BNO055 IMU is utilized to obtain the current orientation as well as the linear acceleration of the robot. This IMU integrates a 3-axis accelerometer, gyroscope, and magnetometer, as well as performing sensor fusion. Onboard the IMU the raw measurements are fused into orientation and acceleration vectors. These vectors are sent to the PC to allow for calculation and estimation of the robot's current and future position for path planning and recovery purposes.

Extended Kalman Filter (EKF). Pinguino makes use of an EKF to fuse the disparate localization data sources. In previous work an EKF was developed but it did not support Pinguino's GPS data. A ROS node was created to process the raw GPS data and convert it into an Odometry message that the existing EKF could read process. In the node the raw GPS data is synced and recombined to output to a new topic of processed GPS data.

Software Strategy and Path Planning

The path planning algorithm that the robot employs is a trajectory rollout sampling algorithm. This algorithm takes sample trajectories from the robots current position and scores them based on how safe each trajectory is and how efficient that trajectory is at moving us closer to the goal point.

Map Generation

The mapping technique employed involves the use of both a local rolling window and a global static window costmap. This allows the robot to have two maps. The global map is static and does not forget obstacles placed on it unless new information about that object is provided. The local map is a dynamic map that is always centered on the robot and does not continue to store obstacles outside of its window. This allows for a large map to be generated which encompases the robot and its goal along with the ability to update and adapt to any changes in operating environment as well as account for moving objects, changes in the traversable path etc.

Goal Selection and Path Generation

The robot's global planner starts by using Dijkstra's path planning algorithm to plan a global path to the goal. The robot then populates the costmap with obstacles around itself. After the local planner uses the trajectory sampling to create a local plan for the robot to utilize in the efforts to adapt to obstacles and other obstructions that may interfere with the original global plan and updates both plans accordingly.

Additional Creative Concepts
Additional creative concepts include the robot's line processing methods. Using contrast to separate the lines from the other parts of the image, a canny edge detection is then run to detect the borders of the lines. The canny is then fed into the probabilistic hough transform algorithm to detect and overlay lines on the image.

FAILURE MODES, FAILURE POINTS AND RESOLUTIONS

Vehicle Failure Modes and Resolutions
Measures are taken to ensure that the robot can recover in event that it becomes stuck and does not think there are any valid paths to take. The main recovery method that the robot uses is a clearing method that involves clearing the local costmap and populating it with new sensor data to correct any errors and to open up the map for the planner to create a new path. This can prove useful in many situations one example being misplaced obstacles on the map blocking off the goal. The recovery methods in place prevent the robot from being permanently stuck in place.

Vehicle Failure Points and Resolutions
Pinguino is also host to a suite of hardware failure point resolutions. In the case of an controls or other electrical malfunction, the robot has employed both software and hardware emergency stops (e-stops). This includes two hardware e-stops, one on each side of the robot in an easy to reach location. It also includes a software based e-stop which is triggered by a button on our long distance controler.

Loose wires have the potential to get tangled within moving parts or to break. This was remedied by using various types of cable management (zip-ties, cable sleeving, etc.) to keep the wires in place. If the signal path containing the emergency stops were to be broken (an e-stop breaks, for example), the robot will default to the stopped mode, engaging the relays. The Arduino Due can then be used to debug the failure point and determine the cause using an analog input. If an electrical component behaves irregularly, the voltage sensing circuit present in the control e-box can be used to determine if a low battery voltage is the cause. If it is, the voltage sensing circuit will return a value below the acceptable threshold. If not, the batteries can be eliminated as a specific cause. If the motors were to stall and pull beyond the amount of current allotted to them within the power distribution system, the 60A in-line fuse will blow and disconnect the motor, or the circuit breaker will trip and disconnect the control e-box entirely. If the Arduino Due issues erroneous motor commands, or if the signal from the Arduino Due does not reach the Sabertooth undistorted, the cyclic redundancy check (CRC), a unique value included with each motor command, will be compared to the command itself, and if an error is found that command will be disregarded.

All Failure Prevention Strategy

To ensure the minimum amount of failure points and modes, the robot has been well-maintained, and all connections verified to be strong and correctly mated. Pinguino was designed to be as modular as possible, so if a sensor or component nears a failing state, it can be replaced or removed to be examined easily. This modularity also allows for tuning of the robot to the environment it finds itself in, further minimizing potential failures.

Testing

In testing Pinguino, the software was run through simulations constructed by the team to examine usability, performance, and to screen for potential failure modes. The hardware was built and tested in smaller pieces, ensuring that each portion behaved as intended before being implemented in the total assembly. Then, once Pinguino was fully assembled, each subsystem underwent testing in a variety of conditions, including a student-created version of the actual IGVC course.

Vehicle Safety Design Concepts

Pinguino was designed with safety in mind. In addition to the numerous safety features integrated in the electrical system as listed above, Pinguino was designed for visibility. The body is mostly transparent to allow for visibility through the robot. LEDs are mounted in visible locations on the robot to show all people nearby the status of the robot at a glance, with large variances in the patterns/colors per each state. The microcontroller is also constantly updating the system with its current operation mode, whether that be an autonomous or teleoperational mode. If any of the autonomous conditions are not met, the motor controller will not accept any new motor commands from the computer. In software, a little increase in the area of all detected objects helps to offset any inaccuracies experienced in the sensor.

SIMULATIONS EMPLOYED

Simulations in Virtual Environment

The simulation tools used to test the robot include Gazebo, a robust and powerful robot simulation tool used to test the effectiveness of Pinguino's software and performance in a similar environment to the IGVC course. The other simulation tool is Rviz. Rviz is used to visualize a variety of different things that happen under the surface, in the Pinguino's software. These are things such as sensor data, transformation trees, costmaps and the current path plan. This is useful for testing and troubleshooting any of these components individually or simulating the robot in its entirety.

Theoretical Concepts in Simulations

The theoretical concepts that are visualized by our simulation software Rviz are important for creating and tuning the navigation stack of the robot. Rviz allows us to view the relationships and transforms between all of the robots sensors and ground plane and visualize our sensor data in relation to the robot. This allows us to configure our sensor suite and position our sensors to achieve optimal inputs for our data processing.

PERFORMANCE TESTING TO DATE

Component Testing, System and Subsystem Testing, etc.

Table 1. System Parameters as Calculated from Known and Obtained Values

Value	Obtained By	Predicted Performance
Speed	Calculated	4.8 mph
Ramp Climbing Ability	Calculated Experimental	19° ~ 30°
Reaction Time	Limited by rate of LiDAR data	~ 50 ms
Battery Life	Calculated (Worst Case) Calculated (Best Case)	~ 0.5 hr ~1 hr
Maximum Obstacle Detection Window	Limited by LiDAR specifications	100m

With all testing done thus far, the gathered values are given in the table above. The speed was calculated with an obtained value for the motor's RPM at a nominal load, and with the known diameter of Pinguino's wheels. The ramp climbing ability was calculated with the known generated torque and with the weight of the robot, however the experiments undertaken so far have currently been challenging those values. Battery life was considered by summing up the total maximum current draw as allowable by the electrical system on Pinguino, then dividing that by the rated capacity of the batteries as given in their configuration. Robot reaction time and its maximum obstacle detection window are both limited by the LiDAR, as it is responsible for both detecting objects and it has the slowest refresh rate of the sensors overall.

INITIAL PERFORMANCE ASSESSMENTS

Overall, the initial performance of the Pinguino platform is in line with team expectations. Performance this year exceeds that of years past, and will only continue to improve as adjustments are made before the competition.