Rochester Institute Of Technology
Multi-Disciplinary Robotics Club
presents

Amos III

TEAM MEMBERS AT COMPETITION

Alan Meekins, Josh Watts, Jeremy Abel, Jack Catalano, Zack Shivers, Stewart Wadsworth, Ross Snyder

TEAM MEMBERS NOT AT COMPETITION

Alex Sojda, Jason Stanislawski, Brian Rosenberger, Chris Kaczmarick, Devin Blau

FACULTY ADVISOR

Dr. Farat Sahin

1.0 Introduction

The Rochester Institute of Technology (RIT) Multi-Disciplinary Robotics Club (MDRC) will be returning to the 16th annual IGVC for the third consecutive year. We are proud to present to you the newest and most capable addition to our family of competition robots: Amos III. In addition to our new robot, we have developed an improved software framework, the Extensible Robot Interface (ERI), which coordinates sensor input and directs actuator output.

2.0 Innovations

MDRC takes great pride in all projects it completes. Amos III, the latest innovative robot developed by MDRC, is no exception to this rule. Amos III is the third generation of the Amos design family and represents a substantial departure from our previous track-driven entrants to IGVC 2006 and 2007. Development on the Amos platform began during the 2007 IGVC competition year as a development and testing system. It was immediately recognized that Amos could be improved to become a viable competition platform. Over the past year Amos has undergone countless revisions and improvements; three completely new chassis designs have been built and tested. Like a powerful race horse, Amos III has been groomed and trained to perfection for entry into the 2008 IGVC.

The Amos III chassis features robust construction and a modular drawer system. Components mounted on the drawers can be quickly slid out for easy access. Amos III can be rapidly adapted for new missions by either modifying the shelf system or removing it altogether and mounting a new sensor or computer package to the mount points. To simplify maintenance, all bolt sizes and types are standardized, reducing the number of tools used to maintain Amos III. With the team's previous background with

track-driven robots, special emphasis was placed on creating proper shock absorption on the drive train using pneumatic tires.

The Extensible Robot Interface (ERI) was developed to simplify sensor interfacing. ERI is a hardware abstraction library written in C++ and built upon free and open source software (FOSS); this provides algorithm developers with a higher degree of freedom and control. A flexible logging system was built into ERI, facilitating field testing of software and logging. These logs can be used for playback and review at a later date. A major problem discovered during the 2007 IGVC was that stale data are treated the same as fresh sensor data; to prevent this problem, ERI timestamps all data being processed by the system, providing an easy mechanism for the detection of stale data and sensor faults.

3.0 Team Structure

Our participation in the last two years of IGVC has lead to a highly effective and innovative task force. This year, a semi-hierarchical team structure helped us operate at maximum capacity and work well in time-sensitive situations. The team itself consists of participants from last year's IGVC as well as new members selected from the incoming freshman class.

3.1 Goals

The team's primary goal was to build a highly modular hardware and software platform well-suited for use in multiple environments and situations. By compartmentalizing the duties of each section of the design team, we were able to achieve modularity in both the hardware and software design and in the actual team structure and process.

3.2 Process

Before the project work commenced, the Project Leader designed a time-line for each major section of the project. By keeping the time-line in an easily accessible, online format, the schedule was reworked to accommodate individual team members' academic commitments. While the time-line was separated for each design sub-team, each team leader was responsible for knowing what the other team leaders were doing. This enabled the entire team to coordinate and synchronize the design of each discrete sub-system into one unified super-system.

3.3 Collaboration

By documenting the design process in real time through the use of a wiki, the team and sub-team leaders could collaborate and generate living documentation as the design was being implemented. As a result, a electronic manual has been created that details all construction and deconstruction necessary for rebuilding the robot.

3.4 Layout

Each design sub-team had a distinct leader and several others working under them. The sub-team leaders worked with each other to ensure that a cohesive design was being created. Each sub-team had a general time-line for their group which was further overseen by the Project Leader. Towards the end of the project, the three sub-teams came together and finalized the hardware of Amos III. With completed hardware, the Project Leader then directed the testing phase.

4.0 Design Process

4.1 Timeline

After the goals of the project were defined, the mechanical and software design teams began creating and documenting their progress on the hardware platform

and ERI, respectively. Simultaneously, small groups were asked to design and document SLaM and behavioral approaches to solving IGVC. However, during the design of ERI, SLaM was abandoned due to time constraints, and a behavioral approach was taken. Each team was given a month and a half to bring forth design decisions affecting interconnected systems. The remainder of the time allotted before IGVC was used to test the systems and buy components to improve Amos III's performance.

4.2 Four-Step Plan

The design process is composed of four stages; each stage describes a time of high concentration on a certain type of progress.

4.2.1 Step One: Training

The beginning weeks of the project were used to train new members. After determining weak points and strengths, workshops were designed with the goal of harnessing existing skills and teaching new ones. Experience has shown that the most effective way to share and propagate skills throughout the club's member base is through these workshops. Through a combination of casual lectures and hands-on learning: the "professor" presents concepts or methods of accomplishing robotics-related tasks, and the "student" is taken into the lab to execute the newly learned skills. This model is very common in academia and is especially emphasized in RIT's overall curriculum structure. Workshop topics include: machining, soldering, embedded development, C/C++ programming, and software design.

4.2.2 Step Two: Research & Brainstorming

Each member drew upon their experience, class-work and outside reading material and discussed ideas with one another. Ideas were then combined and evaluated based on financial and time constraints. Ideas relating to algorithm design were further refined and discussed at length before a final technique was agreed upon.

4.2.3 Step Three: Implementation / Integration

Each system that was conceptualized was implemented and tied together. Each system was tested individually, but not while it is tied to other systems. This stage was by far the most intense. Therefore the most time is allotted here. Some systems (PID control, for example) were developed by more than one group at a time. As such, while each stage is designed separately, every team member is aware of what the other stages are doing. This ensures a robust, cohesive design.

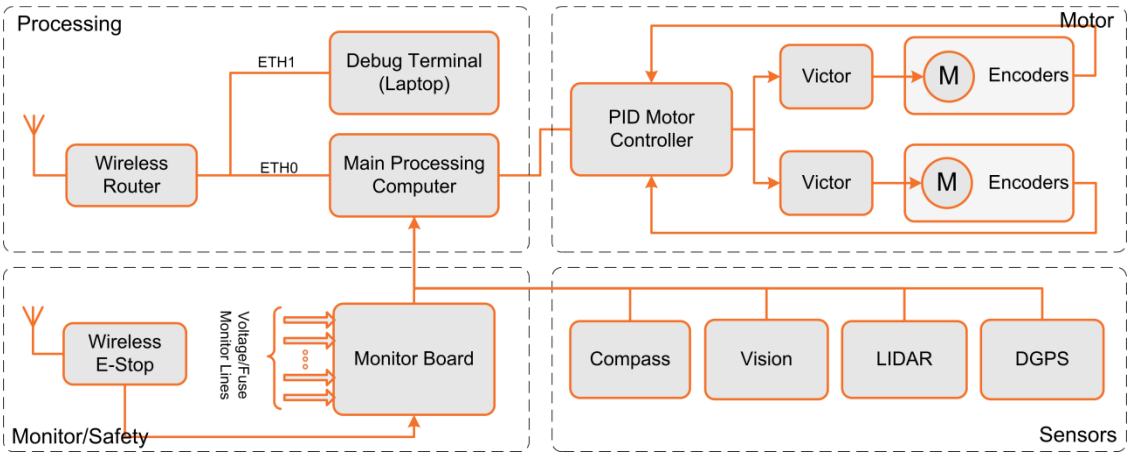
4.2.4 Step Four: Testing

The overall quality of the Implementation Phase was assessed during a final testing phase. A combination of unit, integration, and regression testing was carried out in this phase. These tests were designed to catch implementation problems, assure system interoperability, and finally to snare any bugs that may have been created by system integration. The data logging feature of ERI proved invaluable, allowing testing of algorithms without the need for physical movement of the platform.

5.0 System Overview

5.1 Electrical Systems

During the design phase of the electrical system of Amos III, extensive thought went into striking the right balance between commercial off-the-shelf (COTS) equipment versus custom hardware to optimize performance while reducing overall cost. A potent closed loop motor controller was developed in house to gain the level of precision control demanded by challenges as intricate as IGVC. To ensure rapid transfer of data from the Sick LIDAR, an RS-422 to USB converter was developed in house as well.

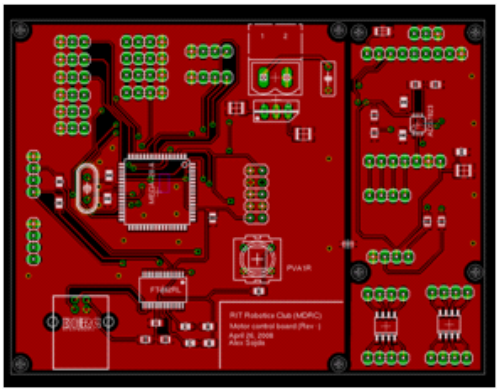


5.1.1 Central Processing Computer (CPC)

The central processing computer (CPC) is a VIA ITX main-board powered by a DC-DC converter optimized for use in 12-volt car systems. This computer features all the necessary I/O to effectively interface to all necessary sensors. Attached to the computer is a wireless router that provides remote wireless access to the CPC for debugging and remote control of the robot.

5.1.2 Motor Controller

Developed in-house, the motor controller is a closed-loop controller, utilizing an ATmega128 8-bit micro-controller to perform PID control of motor velocity. Quadrature encoders provide feedback of the wheel position with a precision of 3.4 mm. The motor controller is connected to the CPC over USB allowing plug and play functionality and simple debugging. The motors are driven using Victor (IFI Robotics) high-current motor controllers. The encoders and embedded control code were all designed and programmed in-house as well.



Motor Controller Board Layout

5.1.3 Emergency Stop

The emergency stop system is vital to the safety of the robot and its supporting personnel, so fail-safe functionality was the most prominent concern. The system utilizes two Arduino micro-controller development boards with XBee wireless radios. There are three situations in which the emergency stop function would be triggered. The first is a wireless stop command thrown manually via the remote. The second condition is thrown when packet integrity drops. This prevents the robot from going out of range and will also trigger when the remote is compromised. The final condition is caused by too many garbage packets on the channel in case of interference from other transmitters.

5.1.4 Digital Compass Interface

A Daventech CMPS03 digital compass is used to determine the orientation of the robot with an accuracy of 4 degrees. It is connected to the main-board through an onboard I^2C connection, with a 5V to 3.3V converter. As a backup, and for testing our system on hardware that does not have onboard I^2C, we have developed a USB interface to the compass using the USB I^2C/IO board by DevaSys.

5.1.5 Camera

Vision is accomplished with a COTS USB webcam. The software interface uses a standard Linux driver (video4linux) and images are processed in a custom-developed video processing ERI module which utilizes Intel's Open Computer Vision (OpenCV) library.

5.1.6 LIDAR

Our robot is equipped with an LMS-291 SICK LIDAR. This unit allows the robot to detect obstacles in a 180 degree arc in front of the robot. The LIDAR is one of the most important sensors on the robot because of its precision and range. Obstacles are detected as far away as 8 meters and are taken into account by the navigation system as soon as they have been classified. In order to communicate at a faster rate, an RS-422 to USB adaptor was fabricated.

5.1.7 GPS

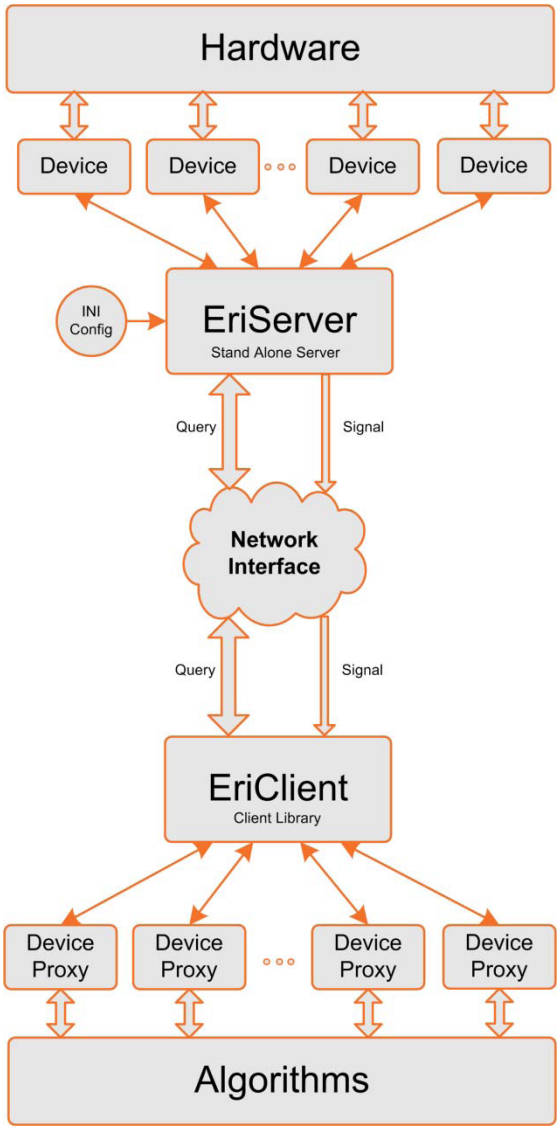
The Global Positioning System (GPS) is used to determine the position of the robot, this is the primary sensor used in the Navigation Challenge. The robot uses an AgGPS132 unit with Omnistar differential corrections. With this capability, the position of the robot is known to within .10 meters. This system connects to the robot through a Linux daemon and a custom ERI module.

5.1.8 Power Distribution / Battery Monitoring System

Effective power distribution is accomplished with safety of the robot in mind. There are two separate battery supplies, each 12V; one is used for the motor and emergency stop systems. The other is used for all electronics (CPC, GPS, LIDAR ...). The LIDAR is passed through a 12V-24V DC-DC converter to achieve the necessary 24V to power the LIDAR. All connections are passed through a fuse to ensure safe operation.

5.2 Software Design

The primary design goal for our software platform was excellent abstraction. Our previous code base, ERC (Extensible Robot Core), was too complex and required in-depth knowledge of each subsystem to create control algorithms. By cutting the connection between hardware protocols and the interface user, system integration has been streamlined with parallel development. The Extensible Robot Interface (ERI) is our solution: it completely abstracts the hardware protocol layer from the algorithm designer, allowing them to focus on solving navigation challenges rather than the oddities of sensor interfacing. ERI was designed to simplify the offloading of data processing tasks by making all sensor data available over a network connection. This means that algorithms built upon ERI can easily share the load of data processing and aid the algorithm developer in harnessing the power of multiple computers.



This is accomplished through a system of device drivers hosted by the EriServer. At startup the EriServer reads an INI configuration file which describes what physical devices are connected. Developers wishing to receive data from a device make use of the EriClient API which handles the details of transferring sensor data from the EriServer. Each device made available by the EriServer is represented by a DeviceProxy in the client API. These DeviceProxies can trigger signals alerting navigation code of the arrival of fresh data which will only be transferred if the data is needed. The EriClient API allows for both event-driven and polled approaches to processing of sensor data making it suitable for many

types of autonomous algorithms. The EriServer can support multiple clients and numerous DeviceProxies, empowering developers to take on larger tasks and challenges.

5.2.1 System Integration

The use of unit testing made system integration a breeze, allowing every system to fall into it place like the carefully machined parts of a Panzer tank. All inter-system dependencies were carefully documented in a process called living documentation which played an important role when the systems were interfaced for the first time.

5.2.2 Image Processing

The image processing subsystem has been extended from the ERC module implemented last year. Last year's design used the Canny Edge Detector and the Hough Transform to segment the image and detect the predominant lines. The system has been extended to track the state of the classified objects between acquired images, allowing the system to determine the robot's pose from external references.

5.2.3 JAUS Implementation

During the 2007 IGVC, the Software Team achieved JAUS Level I compliance and is prepared to demonstrate JAUS Level II compliance this year. Early in the design phase the feasibility of a fully JAUS compliant architecture was explored but was ruled out due to time constraints and the ease with which the required functionality could be added to the ERI framework as a device module.

5.3 Mechanical Design

5.3.1 Drive Train

Amos III's drive train consists of two high-torque DC motors. By using a direct drive system, the speed and torque have been optimized. Since only two moving parts exist in the entire design, the points of failure have been greatly reduced as well.

5.2.2 ◁ Chassis Design

The chassis of Amos III was constructed out of 1/8th inch steel bracing, which provides a very stable yet easily-accessible interior. Diagonal cross-bracing adds additional stability while maintaining an easily-accessible interior. A sloped underside section allows the use of large pneumatic casters while still keeping the chassis level.

5.2.3 ◁ Weight Distribution

The main chassis of Amos III was designed with a very low center of gravity for stability on a wide range of terrain. The center of gravity remains close to the ground due to the lowest possible placement of the batteries. Due to the large amount of space that the batteries and motors take up in the bottom of the chassis, lighter components are placed above on removable shelves.

5.2.4 ◁ Modular Design

The shelving above the main chassis enables the installation of multiple electronic components, computers, and sensors. Various equipment and sensors can be accommodated by repositioning these shelves. The chassis' open frame permits rapid removal of batteries for recharging, thus solving a major problem encountered in the past. The removable top panel gives us easy access to fuse panels and wiring for quick repairs or replacements as well. The position of the LIDAR was carefully chosen as well so as to not create a $5000 bumper.

6.0 Conclusion

6.1 Performance Analysis

Attribute	Design Goal	Final Product
Speed	5 mph	4.5 mph
Reaction Time	Near Instantaneous	100 ms
Battery Life	2 Hours (normal operation)	2 Hours
Obstacle detection distance	8 meters	8 meters
Vehicle performance with obstacles	Perfect	Near Perfect
Navigation point accuracy	.2 meter	1 meter

6.2 Cost Analysis

Part	Vendor	Part Number	Quantity	Cost (actual)	IGVC Cost
Frame Materials (Steel)	Metal Supermarket	-	30ft	$100	$100
Frame Materials (Aluminum)	Metal Supermarket	-	15ft	$75	$75
Misc. Mechanical	-	-	-	$75	$75
LIDAR	SICK	LMS-291		$3000	-
DGPS		AgGPS132	1	$500	-
Digital Compass		CMPS03		$60	$60
Motors		-	2	$500	-
Main-board	EPIA	MII		$150	-
Misc. Connectors		-	-	$40	$40
Solid state storage	Newegg.com	4 Gb Flash Card		$35	-
Webcam	Newegg.com	-	1	$50	$50
Victor Motor Controllers	IFI Robotics	883	2	$149	$149
				$4659	**$549**