I.N.V.A.D.R

2009 Intelligent Ground Vehicle Competition Design Report

Department of Computer and

Electronics Engineering

Peter Kiewit Institute

University of Nebraska

Created By:

Ben Barenz
&
Austin Steiner

Faculty Statement:

I hereby certify that the engineering design in I.N.V.A.D.R by Ben Barenz and Austin Steiner has been significant and equivalent to what is awarded credit in a senior design course.

_____

Dr. Lim Nguyen (Faculty Advisor)

Table of Contents

PAGE

Table of Contents 1

Introduction 2

Team Overview 2

- **Labor Breakdown** 3

Design Process 3

- **MCX Modules** 4
- **Main Control Unit** 5
- **Engine Control Module** 6
- **Intelligent Camera Module** 7

Software Design 7

- **Signal Processing** 8
- **Autonomous Navigation** 8
- **Obstacle Avoidance** 10
- **System Integration Plan** 10
- **Safety** 11
- **Reliability & Durability** 12

Performance and Specifications 12

- **Speed & Ramp Climbing** 12
- **Reaction Time** 12
- **Battery Life** 12
- **Detection Distance** 13
- **Vehicle Specifications** 13

Appendix A – Bill of Materials 14

Introduction:

The following report will describe the I.N.V.A.D.R (**I**ntelligent **N**avigation **V**ision-enabled **A**utonomous **D**igital **R**obot) project developed by Ben Barenz and Austin Steiner. I.N.V.A.D.R was initially designed as an autonomous sentry robot for our senior thesis design course and is intended to be a flexible alternative to standard security solutions. The system is designed on an ATV base, allowing it to be operated in various outdoor environments. An onboard GPS and magnetic compass enable the unit to autonomously drive to pre-determined locations. Once at a given location the unit uses an array of sensors to detect motion, light, and sound mounted on a movable turret system. The platform uses four ultra sonic range sensors mounted at each corner to detect and avoid any obstacles that could be in the drive path. Upon detection the unit will automatically point an onboard camera toward the event area and notify a base station of the event. These features enable the I.N.V.A.D.R to be used as a drop in security solution where there may be no present security infrastructure, such as a construction site. For the IGVC event we removed the turret system which houses all the environmental sensors as these are most pertinent to the sentry features. In order to make I.N.V.A.D.R more equipped for the IGVC event we added a stationary intelligent camera system (CMU2 Cam) and an IR range sensor mounted on a servo for object detection. This new equipment coupled with the existing platform creates a versatile competitor for the IGVC events.

Team Overview:

The I.N.V.A.D.R team consists of Ben Barenz and Austin Steiner who are senior electronics engineering students. The I.N.V.A.D.R project started in fall 2008 in our senior thesis proposal class where we proposed the idea and laid out our initial planning of the design. Actual construction of the I.N.V.A.D.R did not start until January 2, 2009 and the major completion of the project was achieved on April 11, 2009. The final presentation of I.N.V.A.D.R was given on May 5, 2009 when we were officially completed with our senior thesis design course. Preparations for the IGVC event started after this date and continue to present. The project was funded, designed and constructed by Ben Barenz and Austin Steiner. We did not pursue any sponsorship or help of any kind from outside sources. All work was completed at Ben Barenz's work shop at his home utilizing no university resources of any kind.

Labor Breakdown:

Ben Barenz: (Project Manager, Hardware Engineer)

Tasks performed were design and construction of the chassis, steering module, shifter module, rear brake module, front brake module, Throttle control servo and the integration of each in to the chassis. Designed and constructed the complete turret system, performed all welding, cutting, painting of all chassis module and turret components. Ben also designed the control systems for the engine (ECM) and for the turret (LTC & UTC). Designed and installed RGB headlights and installed 55W headlights to front rack. Ben wrote the firmware for the LTC, UTC, Shifter, steering, base station power controller and RGB headlight controller.

Total hours for Ben: 788

Austin Steiner: (Systems Engineer, Software Engineer)

Tasks performed were preliminary testing of software algorithms and design/construction of the main control unit. Once the system was fully constructed and all electronic hardware was in place primary responsibilities switched to programming the main control unit, engine control module, front and rear brake modules, handheld remote control, and base station interface program.

Total hours for Austin: 544

Total hours spent on project: 1,332

The bill of materials for the I.N.V.A.D.R can be seen in Appendix A. All costs were covered by Ben Barenz and Austin Steiner.

Design Process:

Our design for I.N.V.A.D.R started in fall of 2008 when we planned to build a di-cycle platform. After carefully calculating the payload requirements and the capabilities of the available motors in our price range, we determined that we would not be able to build a di-cycle and stay within our budget. Our plan B was to use an ATV platform as we had found a brand new youth ATV that could be purchased from an online supplier for $679 (including shipping). Once we had the ATV we had to determine how to modify the existing platform to meet our design requirements. As we found it difficult to determine exactly what we needed to build for the system we decided it would be best to modularize our system to save on large scale development planning. This allowed us to design individual components ahead of time then fit them into the existing chassis when it came time to install them. This greatly simplified the process of integrating our equipment into the existing platform. For features that required motor control we designed the MCX modules which are described in detail later in this section. The existing magneto (generator), capacitive discharge Ignition module (CDI), and the gear position indicator switches were the only components of the original electrical system that were reused. The rest of the electrical system was designed by the I.N.V.A.D.R team. The system diagram can be seen in figure 1.

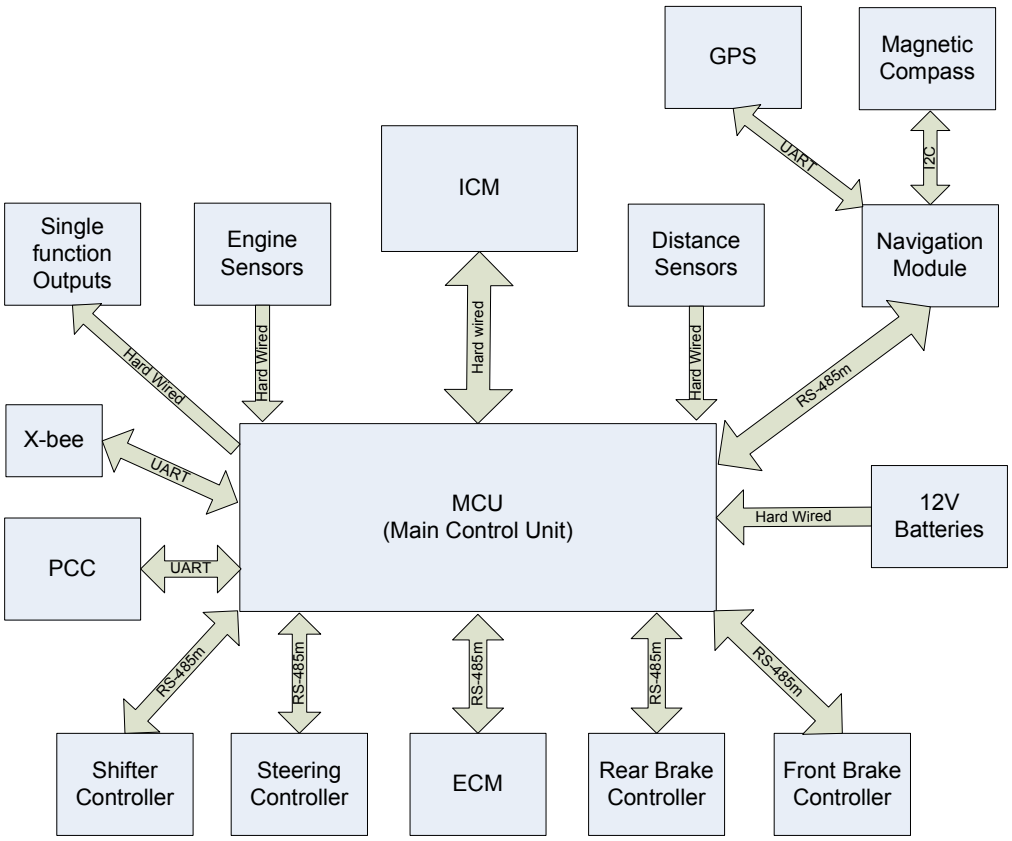

Figure 1 *I.N.V.A.D.R System Diagram*

The system is designed to be modular with individual modules connected to a common serial bus to a main controller. The individual modules handle dedicated tasks such as steering and braking where the main controller is tasked with sending commands to the modules, handling data from the modules, running high level programming, and communicating with the PCC (Processing & Control Console).

MCX Modules:

In order to save cost and system complexity we designed a flexible DC motor controller module that has an on board RS-485m controller called the MCX module (Motor Control Expandable). We designed printed circuit boards and made four of these modules to be implemented as the Front Brake Controller, Rear Brake Controller, Shifter Controller, and the Steering Controller.

The MCX controller uses an ATMEGA48 AVR microcontroller with 4k of flash memory in a QFP-32 package. . We chose this microcontroller as it is the least expensive AVR with a UART and we have extensive experience developing with this particular model. The motor control uses two SPDT relays to form an H-bridge and an N-channel MOSFET tied to the low side of H-bridge to enable speed control of the motor. This configuration was used as it is the most power efficient option and it is less expensive than a full MOSFET H-bridge. The UART lines and a general I/O was dedicated to the RS-485m interface which utilizes a SN75176 RS-485 line converter. All other unused I/O's are tied to headers for future use.

There are three more circuit boards in use on the I.N.V.A.D.R system:

- Main Control Unit (MCU)
- Engine Control Module ECM)
- Intelligent Camera Module (ICM)

Main Control Unit (MCU):

A block diagram of the MCU subsystems can be seen in figure 3.

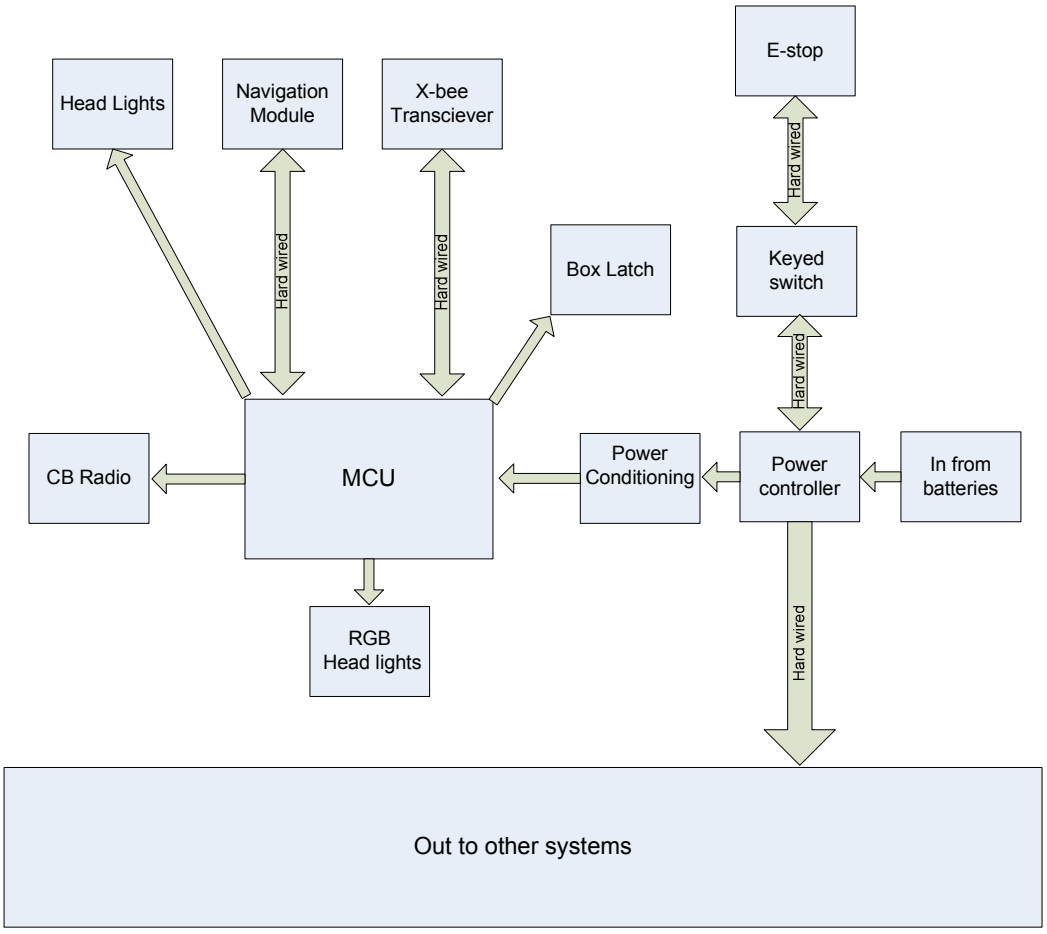

Figure 2 *MCU Subsystem Diagram*

The MCU is tasked with being the "brain" of the robot. It handles all the higher order programming, decision making and pipes signal between all the modules and the PCC. Extra high level processing is shifted on to the PCC when image processing is needed for obstacle detection and avoidance. Under sentry or remote control modes it is not necessary to have the PCC on the I.N.V.A.D.R. The controlling device for the MCU is an ATMEGA 2560 with 256k of flash memory and 86 programmable I/O's in a TQFP-100 package. A full description of this microcontrollers feature can be seen in Appendix B. We chose this microcontroller because we needed 72 general purpose IO pins for our project. The remaining 14 unused pins are available at headers on the board.

Engine Control Module (ECM):

A block diagram of the ECM subsystems can be seen in figure 4.

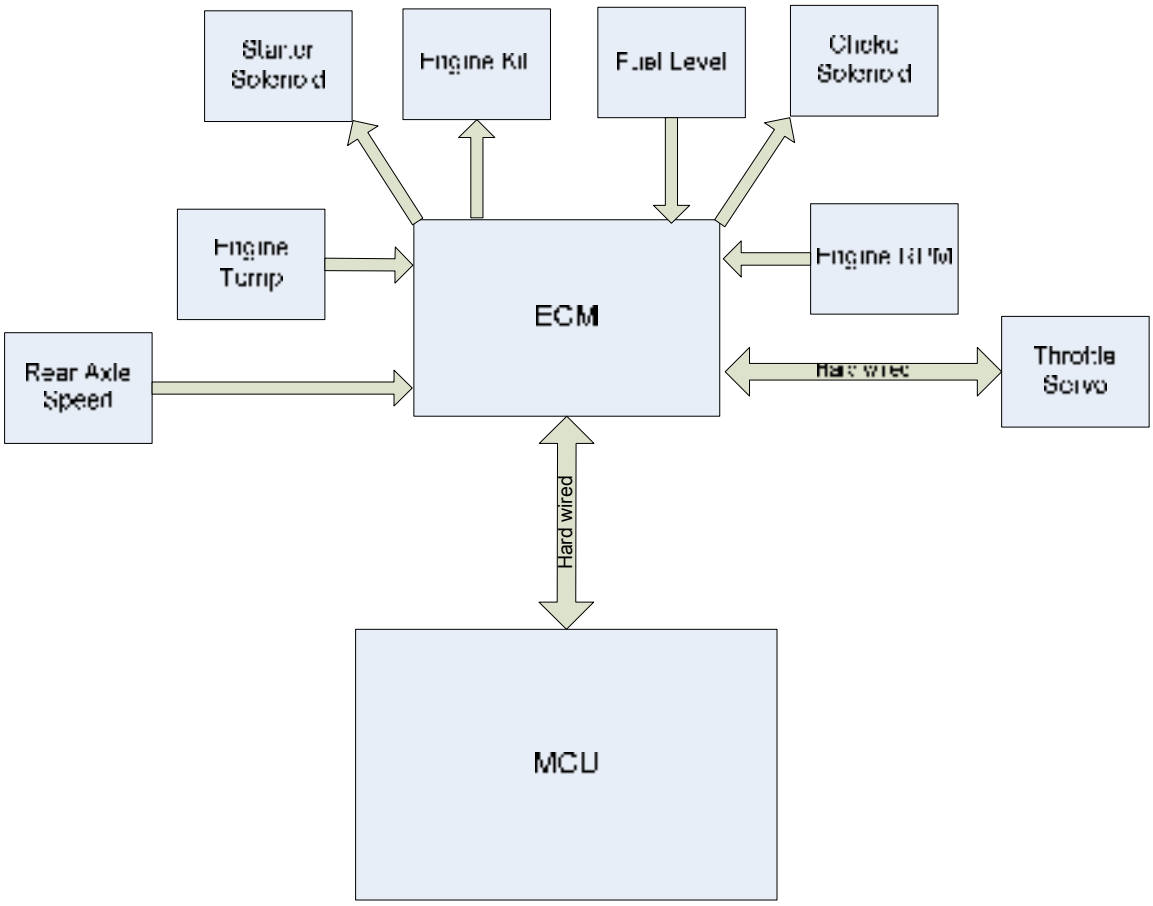

Figure 3 *ECM Subsystem Diagram*

The ECM is tasked with controlling the engine. The signals input/output to the ECM from the engine are:

- Engine RPM
- Engine Temperature
- Choke Actuator
- Fuel Level
- Engine Kill
- Starter Solenoid
- Throttle Control
- Rear Axle Speed

The ECM takes commands from the MCU and uses these I/O's to run the engine. For instance if the MCU sends a command to start the engine it is responsible for running the starter until the engine cranks over.

Intelligent Camera Module (ICM):

The ICM is still in the design phase as it is a new system we are developing for the IGVC event. Currently the ICM consists of a CMU2 camera system which will enable the I.N.V.A.D.R with ability to detect: color, edge, object and blobs. We intend to use this feature to detect the road cones and to detect the lines. This data will be fed back the MCU which will then send it to the PCC for further processing. The PCC will then send drive commands back to the MCU.

Software Design:

For software development we used a combination of Atmel AVR assembly and C. The development environment used was Atmel's AVRStudio4, with the open-source WinAVR C compiler for the C code. By creating a reliable set of routines for handling packet-based communications once we were able to simply copy that code into other modules' source code and be guaranteed the same reliable functionality. The PCC was programmed in Visual Basic .NET using the Visual Studio 2008 development environment.

The main structure of our control code is an infinite loop scanning for and responding to event flags. There are various interrupts running in order to handle all the data traveling over the communications links. There are also numerous events being triggered by a system counter. For example, every two seconds a flag is set to trigger a GPS location read. Whenever a flag is set the main loop immediately (or when it finished handling the current event) clears the flag and runs code to execute the proper response (such as sending data to another device or activating peripheral functions). The architecture of one master loop handling all events guarantees that no two pieces of code are ever fighting for system resources as a response to an interrupt. This is because only the main code can access these shared resources and it can only execute one task at a time.

Software for the sub-systems is similar in methodology. The most significant change is that many of the sub-systems have motors to control. For this purpose these programs have a timed interrupt that checks motor position and issues motor commands to achieve the proper setting of their respective motors.

The software on the PCC is more relaxed with regard to resource sharing because of the more advanced multi-threading tools available in a high level language. These tools prevent problems with resource sharing, eliminating the need for special software consideration.

Another methodology employed in our software is separation of tasks. For example, the navigation module is the only microcontroller where floating point math is used. These routines are not very fast and would use a large amount of processor time on the main controller. For this reason they are separated into a system whose quick response to events is not crucial. Similarly, the tasks of measuring engine RPM and rear axle speed are delegated to the ECM and rear brake module respectively. These are time sensitive measurements and could not be executed by the same processor.

Signal Processing

Most of our distance sensors are simple devices with an analog voltage output. This will be measured with the analog to digital converters on the main microcontroller. Averaging and median select may be applied to help eliminate noise.

Camera signals are processed by an onboard processor (in the case of the CMUcam2) or using a signal processing computer application (such as MATLAB/Simulink).

Autonomous Navigation

The robot begins by sending a target coordinate to the navigation module. The navigation module uses its onboard GPS module and magnetic compass to determine the robot's position and heading. Knowing this and the target location, it can calculate the distance and relative heading to the target coordinate (see figure #4). These two pieces of information are then reported back to the main control unit for use in navigation.

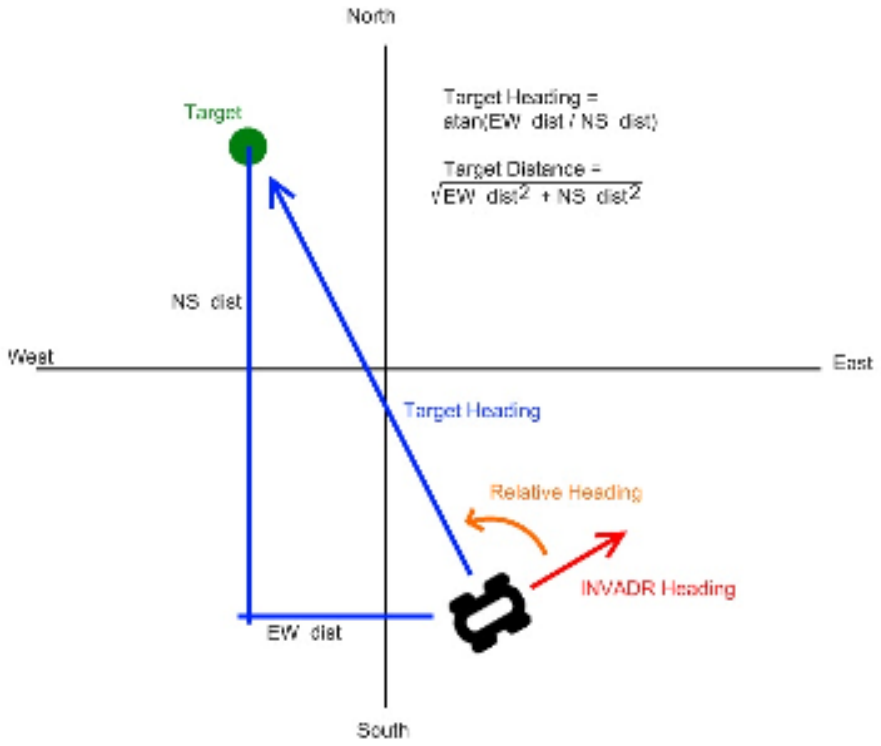

Figure 4 *Navigational Calculations*

Once the robot knows the relative heading to the target coordinate it makes a decision on what direction to steer (see figure #5). This is a proportional control based on the heading error. The area directly behind the robot is defined as a hysteresis zone. In this area the robot will steer whichever way it did last.

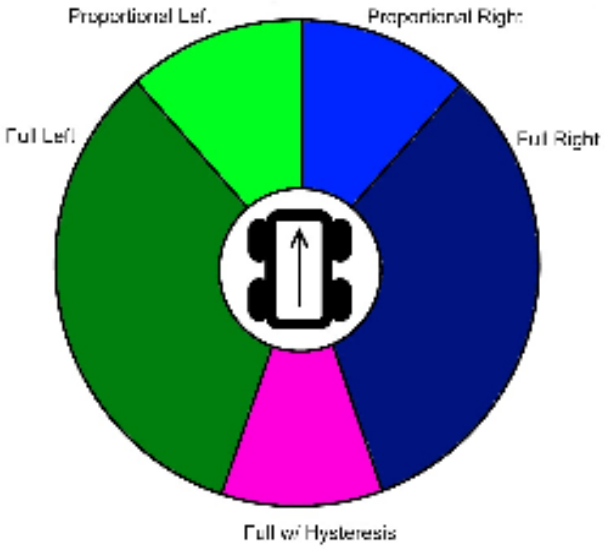

Figure 5 *Proportional Heading*

When the distance to the target is less than 5m, the robot will consider itself at the target and will stop. At this point the onboard computer will upload the next waypoint to the robot and navigation will continue.

While navigating to the target the robot will also be using input from the range sensors to avoid obstacles. The robot will simply steer away from any obstacles, and resume the previously discussed algorithm when the obstacle is no longer detected.

Obstacle Avoidance

The robot has several distance sensors to detect obstacles in its path. It uses image processing from onboard video cameras to detect the lines marking the edge of the path. The top priority for obstacle avoidance is to steer away from obstacles using data from the distance sensors. After this the robot will look for the edge of path lines and try to stay inside them. If neither one of these rules is in effect the robot uses a magnetic compass to steer straight.

Our control algorithm is based on simple rules acting in concert. This will make it more complicated to deal with complex obstacles. We will have to simulate these obstacles to determine the common failure mode and gather data about how the robot can detect them. Each situation will then have to be solved case by case.

System Integration Plan

Standard TTL level serial signals are used to carry the majority of the data amongst the modules of our robot. The different data and signal links are described below.

X-Bee

The X-Bee wireless modules are used to communicate between the handheld remote control, base station, and robot. They communicate to the microcontrollers/computer with a standard serial port and with each other using 900MHz radio.

RS-485m

This is a modified version of the industry standard RS-485 used to enable communications between the different modules on the robot. This type of bus uses differential signaling to place data onto a shared bus. Our system is modified version in that each device has a separate chip select signal. This makes it perfectly clear to the devices which one is being addressed without taking up bandwidth with addressing. It also prevents bus contention by only allowing devices with an active chip select to enter transmitter mode and put data on the bus.

Intelligent Camera Module (ICM) Serial Port

This is a TTL level serial port for sending commands from the main controller to the ICM and sending sensor data back to the main controller.

Processing & Control Console (PCC) Serial Port

An interface is provided to talk to an embedded PC for performing complex calculations.

Engine RPM

The RPM signal from the engine is pulse of about 140V amplitude. This is reduced using a high impedance voltage divider and converted to standard logic levels.

Wheel Speed

The signal from the wheel speed sensors is a current output. The increased current of a pulse is detected and converted to a standard logic voltage signal.

Video signals

The signal from the camera is fed into the PCC via a standard USB port. The data is then captured and analyzed by image processing software (such as MATLAB).

Safety

A number of safety features are implemented both in the hardware and the software of our robot. The first is a hardware emergency stop (e-stop) switch. When this is pressed it shuts down power to the entire robot. No circuit is powered and the engine will stop. A second hardware safety feature we have is a keyed power switch. The power switch must be on for any part of the robot to be on. If it is not the robot can not move any motor or start (or run) the engine. The fact that it is a keyed switch means that nobody but us can activate it.

For software safety all modules implement an e-stop mode. This mode is entered by reception of a certain packet that is broadcast to all devices. The specific actions taken by the modules depend on their function, but the end result of e-stop mode is that the brakes apply fully to bring the robot to a complete stop and the engine shuts off. This mode must then be cleared manually by the operator before normal functionality is restored. An e-stop can be triggered from either the PCC or the handheld remote control. In normal operation the PCC and handheld remote periodically send keep-alive packets to the robot (either of which can be disabled). If the robot ever goes out of communications for more than four seconds the robot will execute an emergency stop.

Reliability & Durability

Our system uses no non-standard components. As such the reliability of the electronics is as specified by the individual manufacturers. The boards were manufactured using industry-standard PCB manufacturing methods and are very reliable. Plastic and metal enclosures protect all electronic parts from the outdoor environment. All cabling and mounting of electronic parts was done using industry standard connectors that are made to withstand the environment of a functioning robot (vibration for example) The mechanical parts of the system are built using commercial products designed to be driven in rough outdoor environments. All modifications done to the original hardware were done with reliability and ruggedness in mind and match or exceed the durability of the original hardware.

Performance and Specifications:

Speed & Ramp Climbing

As a commercial ATV our robot could exceed a speed of 40mph. It could also carry a payload of over 300lbs. and pull a trailer uphill. Since we have made no modifications to the fundamental ATV systems it can still meet these values. We have not driven it at 40mph via remote control for safety reasons. It has easily driven over dirt piles with an incline of about 45 degrees. Our target speed for navigation at the IGVC is about 5mph.

Reaction Time

The reaction time of our robot is limited by the time it takes for the DC motors to move to their commanded positions. The maximum travel time for any motor is about one second (for the brake motors). There is also a reaction time restriction because the X-Bee wireless modules we are using for communications at times buffer data for a few seconds then send it in a burst. This is usually not a problem but can cause a small delay in control.

Battery Life

Our vehicle is powered by gasoline and an internal combustion engine. The battery pack has a capacity of 70Ah and is only used to power the DC motors that actuate controls. We estimate average current draw to be about 2A, which gives us about 35 hours of run time (ignoring running out of gas). It would take the engine about 6 hours to burn all the fuel from a full gas tank. Without fuel the electronic systems of the robot could still operate, but the engine could not run.

Detection Distance

The ultrasonic sensors we use have a useful range of 20.2ft. The infrared range finder has a range of about 18ft. The range of the cameras is only limited by how they are mounted and their field of vision. We have not tested any of these sensors on the robot but have had good results with these sensors in past projects.

Vehicle Specifications

Engine

- 125CC Air cooled, 4 stroke, single cylinder, 6.5Nm/5000rpm
- Electric start, Electronic Ignition CDI

Transmission

- Automatic, Forward, Neutral, Reverse (with position feedback)
- Chain Drive

Chassis Specifications

- Tires size 16” X 8” X 7”, Ground Clearance 3.54”
- Wheelbase 32”, over all size 50” X 33” X 38”
- dry weight 315lbs, Max Load 400lbs
- Brakes: Front Drum, Rear Disc
- Fuel capacity: 1 gallon

Electrical Specifications

- Voltage 12VDC, 70aH capacity
- Magneto output: 12V, 1A @ 2000rpm
- External charger input

Appendix A

Bill of Materials

I.N.V.A.D.R BOM

I.N.V.A.D.R						
Chassis						**Comments**
Description	**Vendor PN**	**Vendor**	**Qty./Unit**	**Cost**	**Extended**	
Youth ATV	ATA 125D	SL Powersports	1	$679.00	$679.00	
5/8" 2 Bolt Flange Bearing	1-202-10-2	Surplus Center	3	$7.95	$23.85	Turret and Steering Bearing
U1 12V 35Ah Battery	N/A	Ebay	2	$77.50	$155.00	Batteries
Tilt Motor	5-1571	Surplus Center	1	$11.95	$11.95	
Pan Motor	5-1257	Surplus Center	1	$12.95	$12.95	
Steering Motor	5-1648	Surplus Center	1	$119.95	$119.95	
Throttle Motor	5-1587	Surplus Center	1	$2.99	$2.99	
MCX motors	5-1664	Surplus Center	4	19.95	$79.80	
25 pitch gears and chain for Pan	N/A	Surplus Center	1	$25.00	$25.00	
55 Watt headlightas	93904-5VGA	Harbor Frieght	2	$7.99	$15.98	Head lights
Gas Tank	28-1452	Surplus Center	1	$9.95	$9.95	
Fuel Filter, Line, valve	N/A	Auto store	1	$9.95	$9.95	
Various Wiring, Cable ties, etc	N/A	Various	1	$15.00	$15.00	
Choke Solenoid	470008	Surplus Center	1	$2.69	$2.69	
Various other Hardware	N/A	Various	1	$50.00	$50.00	
Brief case	N/A	Harbor Frieght	1	$26.00	$26.00	Base Station Case
Steel	N/A	State Steel	1	$50.00	$50.00	Chassis Metal
			Chassis Total		**$1,290.06**	
Electronics						**Comments**
Description	**Vendor PN**	**Vendor**	**Qty./Unit**	**Cost**	**Extended**	
Ultra Sonic Range Sensor	Maxbotix LV-EZ3	Spark Fun	5	$27.95	$139.75	IR Analog Distance Sensor
Light Sensor	LIGHTSENSOR	Futurlec	7	$0.65	$4.55	Linear Light Sensor
PIR Sensor	PIR_MODULE	Futurlec	6	$6.90	$41.40	
Humidity & Temperature Sensor	SEN-08257	Spark Fun	1	$41.95	$41.95	
MOSFET N-CH 55V 100A TO-220AB	HRF3205-ND	Digikey	4	$1.83	$7.32	MOSFET transistor
SHARP 1/3" CCD CAMERA	N/A	Ebay	1	$50.90	$50.90	Turret Cam
AV Transmitter/Receiver Kit	N/A	Ebay	1	$74.98	$74.98	AV radio
Compass IC	HMC6352	Spark Fun	1	$60.00	$60.00	
GPS	EM-406a	Spark Fun	1	$60.00	$60.00	
Circuit Boards	N/A	Advanced Circuits	3	$33.00	$99.00	
X-bee9 900 Mesh Dev. Kit	XBP09-DMDK	Digi	1	$250.00	$250.00	Wireless Link
SN754410	595-SN754410NE	Mouser	2	$1.65	$3.30	H-bridge
Atmega 2560	556-ATMEGA2560-16AU	Mouser	1	$16.77	$16.77	MCU
SN75176BP	595-SN75176BP	Mouser	10	$0.72	$7.20	RS-485 IC
Pressure Sensor	MPVZ5010G	Mouser	1	$12.30	$12.30	Fuel Pressure sensor
P-Channel Fet	FQPF12P10	Mouser	6	$0.84	$5.04	Drivers
Switching regulator Controller	863-LM2576T-005G	Mouser	2	$2.49	$4.98	UTC & MCU supply
CB Radio	Midland 1001Z	Ebay	2	$40.00	$80.00	CB radios
Green Laser	N/A	Ebay	1	$18.96	$18.96	Green laser
Base Station battery	N/A	Ebay	1	$18.95	$18.95	Battery for base statoin
CB adapters	N/A	Ebay	1	$19.85	$19.85	coax adapters
Two 10in Rubber CB attenna	N/A	Ebay	1	$18.60	$18.60	CB attennas
Shielded CB cable	N/A	Ebay	1	$8.99	$8.99	CB cable
Atmega 48	556-ATMEGA48P-20PU	Mouser	8	$1.20	$9.60	MCX modules
Attiny24	556-ATTINY24-20PU	Mouser	2	$1.91	$3.82	RGB lights
Various other Electronics	N/A	Mouser	1	$30.00	$30.00	
			Electronics Total		**$1,088.21**	
			Grand Total		**$2,378.27**	