Intelligent Ground Vehicle

Jihang Li, Pierre Larochelle, Shashank Bishnoi

Robotics & Spatial Systems Laboratory
Department of Mechanical and Aerospace Engineering
Florida Institute of Technology
Melbourne, Florida 32901

jihang2014@my.fit.edu, pierrel@fit.edu

ABSTRACT

This article presents the progress of the Intelligent Ground Vehicle (IGV) project, which is trying to implement the Simultaneous Localization and Mapping (SLAM) technique to a three-wheeled mobile robot. A well designed and prefabricated autonomous mobile robot research platform has been provided by previous works and is discussed in this article to demonstrate the prospection of this project.

Keywords: mobile robot, outdoor, autonomous, Intelligent Ground Vehicle, SLAM

1. INTRODUCTION

1.1 Project Overview

The Intelligent Ground Vehicle (IGV) project began as a senior design project of which the goal was to create an autonomously navigated vehicle to be entered into the Intelligent Ground Vehicle Competition (IGVC), which is an engineering challenge that the main goal is to design a robot with artificial intelligence to autonomously navigate through a course full of obstacles such as barrels, pot holes, and sand traps. The robot is equipped with a SICK LMS211-S07 laser range finder (LiDAR), a digital compass, a Hemisphere Eclipse digital global positioning systems (GPS), a stereo camera, two servo drives with encoder feedback, and a 24V (DC) supply for up to 2 hours of autonomous operation.

Instead of building a mobile robot that runs on a specific field, right now this project is being modified in order to be more generally functional so that at the end of this project the IGV should be able to navigate through outdoor environments to a designated destination with a given GPS waypoint coordinate only.

1.2 Related Works

As machines that are capable of locomotion, mobile robots have been studied and developed for a long time. With an early history starts with William Grey Walter's tortoise robots in 1940s, mobile robots have become more commonplace in their commercial and industrial versions nowadays, such as vacuum cleaners, robot dogs, quadcopters and industrial warehouse conveyors[5]. As for IGVC, a lot of teams have successfully built their own robots to enter the competition and some of those have finished the challenges in this competition since 1993, the first year of IGVC [6]. Furthermore, the autonomous car ”Stanley” that created by Stanford University's Stanford Racing Team in cooperation with the Volkswagen Electronics Research Laboratory (ERL) has proved that the outdoor SLAM problem has been intensively studied and has been answered in some way [11].

2. PROBLEM SPECIFICATIONS

The IGV aims to reach a given GPS destination in finite time, while running in outdoor environments containing different kinds of terrains, stationary obstacles and moving objects. To successfully finish the mission, IGV should have good mechanical structure robustness and enough artificial intelligence to navigate through the field. More specifically, this project has three main challenges: hardware design challenge, autonomous challenge, navigation challenge.

2.1 Environment Specification

Since IGV is an outdoor oriented robot, the chassis should be able to prevent the electronic system from direct contact with sunshine and light drizzle, in case of being overheated or short-circuited (Table 1). In addition, the robot has to be capable of running on a grass field or on concrete paths (Table 2).

Table 1. Weather Specification

Weather	Specification
Temperature	Minimum: $10°C$ Maximum: $40°C$
Radiation	Be able to operate in full sun
Rainfall	Be able to operate in light drizzle

2.2 Runtime Specification

The runtime specifications are derived from the estimated time that the vehicle must operate during one run.

Table 2. Terrain Specification

Terrain	Specification	
Grass Field	Grass Height	10 cm maximum
	Discontinuity	10 cm maximum
	Inclination	20° maximum
Concrete Path	Width	1.5 m minimum

Table 3. Runtime Specification

Specification	Requirement
Distance per run	$100 - 200$ m
Battery requirement	Two 12 V, 18 Ah
Runtime without battery swap	30 min continuous
Runtime with battery swap	90 min continuous
Power consumption while in motion	24 V, 30 A average
Power consumption while stopped	24 V, 5 A

2.3 Vehicle Specification

Physical specifications of the vehicle are based on the electronic system architecture. And the navigation accuracies are goals that IGV should achieve during the maneuver, which means the vehicle has to be accurate enough to stay on the optimal path to finish a mission in the shortest amount of time, and ensure itself to stop within a circle around the target point with a radius of 0.5 m.

Table 4. Vehicle Specification

Specification		Requirement
Dimensions		$72 \times 92 \times 92$ cm
Payload capacity	Maximum load	100 kg
	Maximum size	$20 \times 45 \times 45$ cm
Speed		$0.1 - 10$ KPH
Heading accuracy		0.1°
Waypoint accuracy		± 0.5 m

3. CURRENT DESIGN

3.1 Mechanism

To deal with the outdoor environment, the chassis was made from square aluminum tubing (in Figure 1), since that aluminum has a very high strength to weight ratio. There are three different levels in the chassis, which allow the chassis to accommodate all of the components that it needs to transport. The lowest level holds the batteries which will be the bulk of external weight placed on the chassis. The middle level will hold all of the electronics such as motor drivers. The highest level will hold the computer which will be used to create the map and path plan for the robot. Both the second and third levels of the chassis are supported by silicon gel mounts, which mean to alleviate vibrations and prevent the connections of the electronics from coming loose. Additionally, to deal with the specified environment problems, a good enclosure is applied to the chassis to keep the electronic system away from the external environment (Figure 2) and three rubber wheels with a diameter of 0.36 m are used to deal with different terrains (Figure 1, 2).

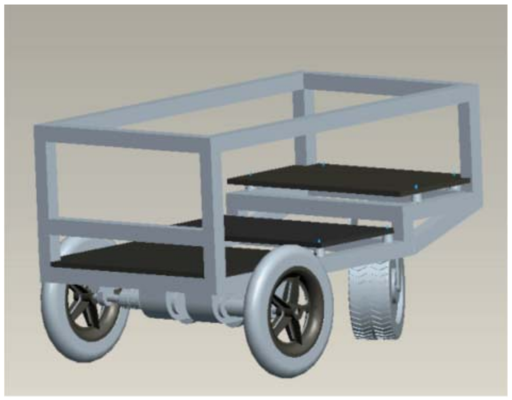


Figure 1. Chassis

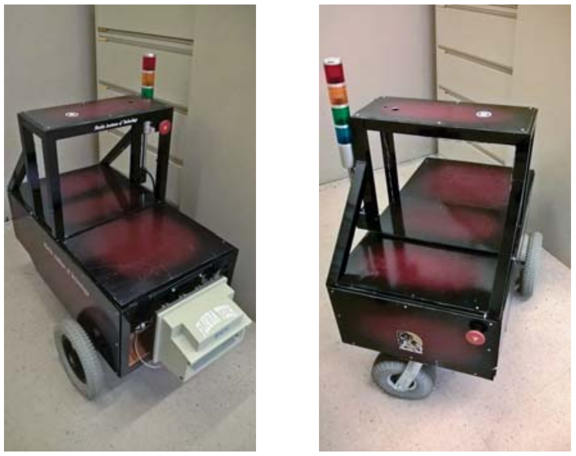

(a) Front View (b) Rear View

Figure 2. Chassis Enclosure

3.2 Motion System

A three-wheel design generates the motions of IGV, with two front independently controlled wheels that used to propel the robot and a back wheel which is free to move in any angular direction. In this case, the robot is steered by using the differential speeds of the two powered wheels, and it will be able to achieve a very small turning radius with the freely turn wheel. Furthermore, two DC motors are used with the front wheels which were used in the design of a commercial wheelchair. All the mentioned components can be seen in Figure 3.

The motor drivers used to control the DC motors are two identical AMC DPRALTE-060B080 servo drives (Figure 4, taken from [2]). These drives are able to operate in torque, position, or velocity mode and feature programmable analog and digital interfacing methods.

Specifically, the command source for velocity mode can be analog input(continuous DC), step and direction, and interface input(serial command). The interface input method is preferred in our case, since no external signal generator is needed. And serial commands will be sent by the onboard PC by RS-485 protocol, which are in a specific form defined by the communication manual[1] provided by the

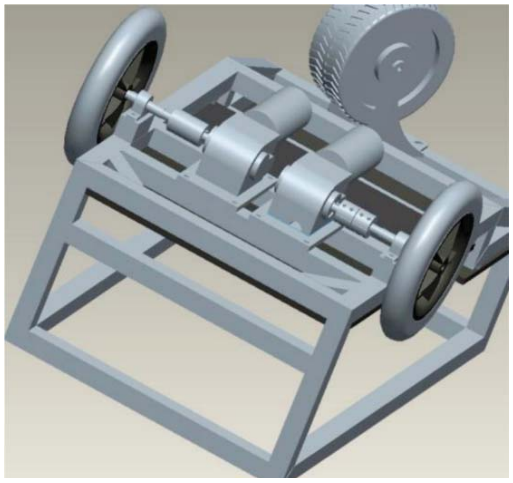
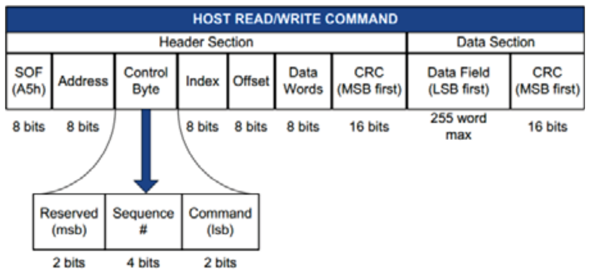


Figure 3. Motion System

Figure 4. AMC DPRALTE-060B080 Servo Drive

manufacturer as shown in Figure 5. Basically, only four commands, representing in hexadecimal form, are required to drive or stop a motor which are "get write access", "enable bridge", "disable bridge" and "set target velocity". The first three commands and the header section of the velocity setting command can be preset with a given network address and a interface port of the servo drive. The only varying part is the data section of a velocity setting command, which consists of a hexadecimal target velocity of least significant bit (LSB) first and a cyclic redundancy check (CRC) particularly calculated according to the target velocity.



HOST READ/WRITE COMMAND								
Header Section							Data Section	
SOF (A5h)	Address	Control Byte	Index	Offset	Data Words	CRC (MSB first)	Data Field (LSB first)	CRC (MSB first)
8 bits	8 bits		8 bits	8 bits	8 bits	16 bits	255 word max	16 bits

Reserved (msb)	Sequence #	Command (lsb)
2 bits	4 bits	2 bits

Figure 5. Serial Command Frame

An actual generation of motion can be described as the following flowchart (Figure 6). The program starts with taking in a target velocity with a unit of "rounds per minute (RPM)", and converts it into a signed 32-bit hexadecimal LSB array with a unit of "counts per second", which means the counting of the lines of encoder. The next step is to calculate a related CRC by using the velocity array and store the CRC into another array. These two arrays then, in the form of "| velocity array | CRC array |", construct the data section of a serial command. Appending this data section to the preset header section and one serial command is constructed and is ready to be sent. More details about the motion generation software architecture and examples of the application in practice will be presented in Appendix A.

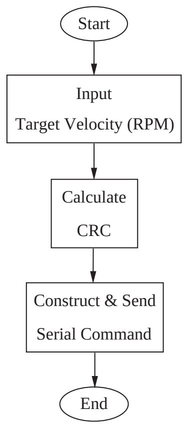


Figure 6. Motion Generation Flowchart

3.3 Sensor System

The selection and application of sensors are based on the navigation problems which can be addressed as "where am I?", "where do I want to go?" and "how do I get there?"[12]. Moreover, concerning the ability of a robot to build a map of the terrains it traverses, a question "where have I been?" is also asked[8]. These together are known as the simultaneous localization and mapping (SLAM) problem.

The sensor system of IGV consists of a LiDAR, a GPS, a digital compass and a stereo camera. However the sensor system has not been finished yet, a blue print is presented in this section.

1. *LiDAR*. The SICK LMS211-S07 LiDAR (Figure 7) mounted on the front of IGV is an outdoor range finder with an operating range up to $80m$ and a $100°$ field of view. The response time can be as small as $13ms$ with a typical systematic error of $\pm 35mm$ and statistical error of $\pm 10mm$[10], which will provide real-time and high accurate data of the circumstance.

2. *GPS*. A GPS will be used to receive the current position coordinates, which will tell the main program "where I am", to verify if the robot is on the designated path or already reach the destination.

3. *Digital compass*. A digital compass can read the yaw, roll and pitch angles of the IGV that defined in a right

Figure 7. SICK LMS211-S07 LiDAR

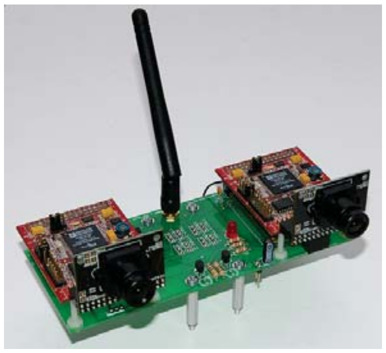
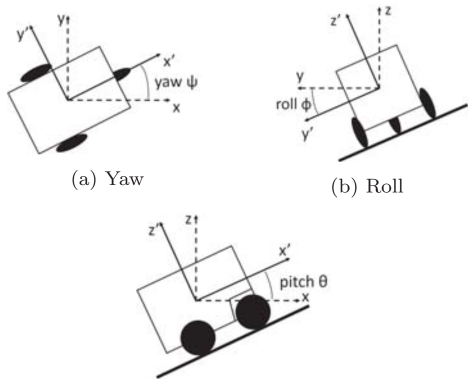


Figure 9. Surveyor Stereo Vision System

handed coordinate system (Figure 8). The pose data can be used to cooperate with GPS while navigating, or to implement dead reckoning navigation when the GPS signal is weak.



(a) Yaw

(b) Roll

(c) Pitch

Figure 8. IGV Pose

4. *Stereo camera.* A Surveyor Stereo Vision System (SVS) (Figure 9) is ready to be studied, which provides a 3D stereo vision and has been implemented into several projects [4][9]. More generally, stereo cameras have been used to apply real-time motion detection [3] and people detection [7].

4. FUTURE WORK

For now the motion of IGV is measured by RPM, which may not be efficient while turning around a corner. Instead, implementing a turning radius (Figure 10), that translated from the difference between two wheels, maybe much easier for further research.

$$x_1(m) = v_1(m/s) \cdot t(s) = r_1(m) \cdot \theta(rad)$$
$$x_2(m) = v_2(m/s) \cdot t(s) = r_1(m) \cdot \theta(rad)$$

The next step of this project will be setting up the sensor system and make it cooperate with the motion system to construct a complete mobile robot, which then should be

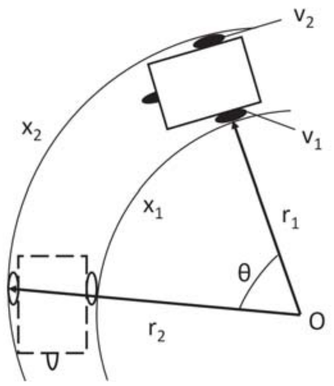


Figure 10. Turning Based on Radius

followed by studying the path planning and map building algorithms. At the end of this project, the SLAM problem needs to be well answered (Figure 11).

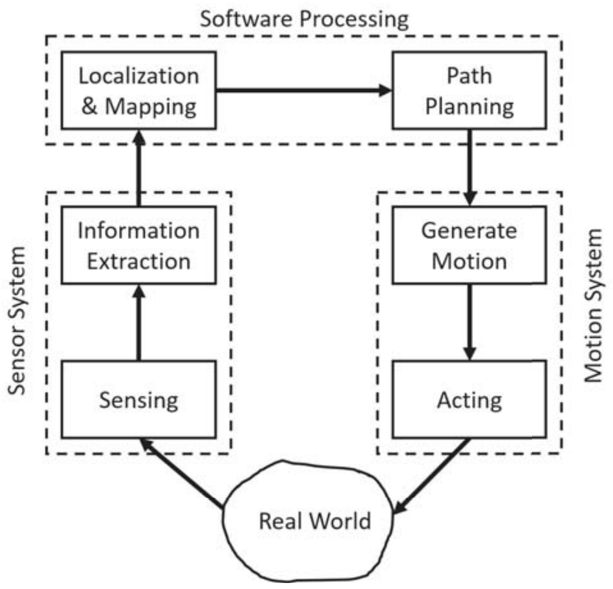


Figure 11. SLAM System

5. SUMMARY

An idea of an intelligent ground vehicle that aims at solving outdoor SLAM problem is proposed in this paper, based on a well designed mechanism, a ready-to-use motion system and several successful related works. What being presented in this paper is an unfinished system, the sensor system and high level data processing algorithms have to be intensively studied in the following works in order to well solve the SLAM problem.

6. ACKNOWLEDGMENTS

We express our sincere appreciation to the Florida Tech Intelligent Ground Vehicle Competition senior design team: Alex, Eric, Michael, Osarhieme, Saleh, and Shashank.

References

[1] Advanced Motion Controls, Feb. 2014. http://www.a-m-c.com/download/sw/dw7-2-2/AMC_RS485_232_CommunicationManual_7-2-2.pdf[Accessed: Apr. 2016].

[2] Advanced Motion Controls, Aug. 2015. http://www.a-m-c.com/download/datasheet/dpralte-060b080.pdf[Accessed: Apr. 2016].

[3] M. Agrawal, K. Konolige, and L. Iocchi. Real-time detection of independent motion using stereo. *Application of Computer Vision*, 1:207–214, Jul. 2008.

[4] H. Gordon, Oct. 2008. http://diydrones.com/profiles/blog/show?id=705844%3ABlogPost%3A48082[Accessed: Apr. 2016].

[5] E. Guizzo. Three engineers, hundreds of robots, one warehouse. *IEEE Spectrum*, Jul. 2008.

[6] IGVC. http://www.igvc.org/teams.html[Accessed: Apr. 2016].

[7] R. Munoz-Salinasa, E. Aguirreb, and M. Garcia-Silventeb. People detection and tracking using stereo vision and color. *Image and Vision Computing*, 6:995–1007, Jun. 2007.

[8] R. R. Murphy. *Introduction to AI Robotics*. MIT Press, Cambridge, Massachusetts, 2000.

[9] S. Rainwater, Sep. 2008. http://robots.net/article/2645.html[Accessed: Apr. 2016].

[10] SICK, Aug. 2013. http://sensorstrade.com/media/pdf/33761/sick-lms211-s07-datasheet.pdf[Accessed: Apr. 2016].

[11] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, and P. Mahoney. Winning the darpa grand challenge. *Journal of Field Robotics*, 2006. accepted for publication.

[12] G. Zunino. Simultaneous localization and mapping for navigation in realistic environments, 2002.

APPENDIX

A. DETAILS ABOUT MOTION GENERATION

In the motion generation program, which is coded in C++, the procedure from a signed integer type velocity input with the unit ”RPM” to a serial command that can be accepted by the AMC servo drives is quite straight. It can simply be achieved by putting all the functions into one file. However, to make it much easier to be debugged and called, the program was meant to be separated into several files and each one has been developed into a ”class” according to its specific responsibility (Figure 12).

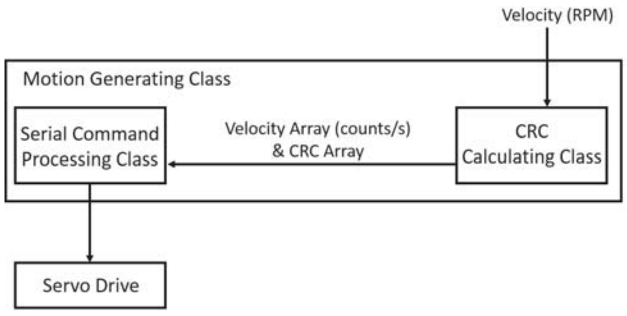


Figure 12. Motion System Software Architecture

With these classes, the target velocity can be set and sent to the servo drive by only two lines of codes:

$$GenerateMotion\ IGV;$$
$$IGV.setVelocity(targetvelocity);$$

Take $targetvelocity = 10$ as an example. Firstly, the target velocity value 10 is sent into the ”CRC Calculating” class and converted into a 32-bit hexadecimal LSM value $B5010000h$ (437). There is a implicit transformation since the servo drives can only understand the counts of the lines of the encoder instead of the rounds that the motor rotates. Such transformation is given by:

$$10\ rev/min * 400\ counts/rev * 1\ min/60\ sec * \frac{2^{17}}{20000\ Hz}$$
$$= 436.9067$$
$$= 437$$
$$= 000001B5h$$

This value will then be used to calculated the CRC, which should be $7AA4h$ in this case. These two results will be store in to two arrays and construct the data section of the serial command.

Next step is to construct a complete serial command according to the command frame defined by AMC serial communication manual as shown in Figure 13, which is also mentioned in section 3.2.

1. SOF. ”SOF” is short for ”start of frame” and is always $A5h$.

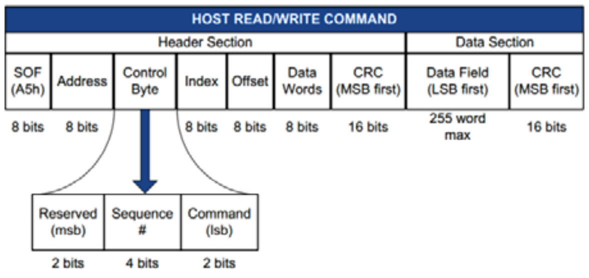
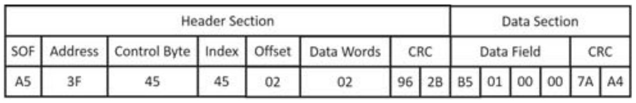


Figure 13. Serial Command Frame

2. $Address$. Indicating the network address of the servo drive, which is $3Fh$ (63) by default and can be customized from $01h$ (1) to $3Fh$ (63).
3. $ControlByte$. $01h$ for ”read”, $02h$ for ”write”. Only $02h$ is used in this project up to now.
4. $Index$. In this example, the ”Index” will be set to $45h$ which means the interface input method is being used.
5. $Offset$. With the ”Index” specified as $45h$, this parameter indicates which interface port is being used. There are 8 available ports which are $00h$, $02h$, $04h$, $06h$, $08h$, $0Ah$, $0Ch$, $0Eh$. in this example, $00h$ is used.
6. $Data\ Words$. This is a value that indicates the number of words (2 bytes) in the data field. Since the data (velocity) is a 32 bit number, ”Data Words” is set to $02h$ here.
7. CRC ($MSB\ first$). Up to now, the header section consists of ”$A53F02450202$”, and this CRC is corresponding to the header section. Thus the CRC here is $962Bh$. The calculating program is given in Appendix B.
8. $Data\ field$ ($LSB\ first$). Set to $B5010000$ in this example.
9. CRC ($MSB\ first$). CRC calculated according to $B5010000$, which should be $7AA4$.

Therefore, a serial command is constructed (Figure 14).



Header Section								Data Section					
SOF	Address	Control Byte	Index	Offset	Data Words	CRC		Data Field				CRC	
A5	3F	45	45	02	02	96	2B	B5	01	00	00	7A	A4

Figure 14. Serial Command Example

B. CRC CALCULATING ALGORITHM

```
usigned int crc_accumulator = 0;
usigned int crc_poly = 0x0810;
int highest_bit;
int CrunchBuffer;

void CrunchCRC(int Crunch_input) // Compute CRC using Bit-by-Bit method
{
    CrunchBuffer = Crunch_input;

    for(int i = 0; i < 8; i++)
    {
      highest_bit = (CrunchBuffer >> 7) & 1;

      if(crc_accumulator & 0x8000){
        crc_accumulator  = ((crc_accumulator ^ crc_poly) << 1) + (highest_bit ^ 1);
      }
      else{
        crc_accumulator = (crc_accumulator << 1) + highest_bit;
      }

      crc_accumulator &= 0x0ffff;
      CrunchBuffer <<= 1;
    }
}
```