DESIGN AND DEVELOPMENT OF A MOBILE ROBOT PLATFORM FOR AUTONOMOUS GROUND VEHICLE RESEARCH

Jihang Li, Pierre Larochelle
Robotics & Spatial Systems Laboratory
Department of Mechanical and Aerospace Engineering
Florida Institute of Technology, Melbourne, FL, U.S.A.
Email: jihang2014@my.fit.edu; pierrel@fit.com

ABSTRACT

The design and development of a mobile platform to support autonomous ground vehicle research is presented. The focus of this paper is on the design and development of Robot Operating System (ROS) based driver libraries for the control and navigation of the robot. Prior works have provided the mechanical system and integrated several sensors for the robot. The current working electronic system includes two DPRALTE-060B080 servo drives for driving DC motors and acquiring odometry, an OS5000-S IMU for reading robot orientation, and a LMS291-S05 LiDAR for range finding. The libraries for the servo drives and IMU are specifically written based on ROS, while a general library for LiDARs that was developed by the ROS community has been utilized. The result is a working platform that supports research in autonomous ground vehicles; e.g. for conducting experiments to study the effectiveness and efficiency of novel algorithms for addressing the so called Simultaneous Localization and Mapping (i.e. SLAM) problem.

Keywords: Autonomous Ground Vehicles; AGVs, ROS.

RÉSUMÉ

La conception et le développement d'une plateforme mobile pour participer à la recherche des véhicules terrestres autonomes sont présentés. Cet article porte sur la conception et le développement de bibliothèques sous ROS (Robot Operating System) pour le contrôle et la navigation du robot. Les travaux précédents ont fourni le système mécanique et ont intégré plusieurs capteurs au robot. Le systeme électronique fonctionnant actuellement comprend deux servo-commandes DPRALTE-060B080 pour actionner les moteurs à courant continu et acquérir l'odométrie, une centrale inertielle (IMU) OS5000-S pour lire l'orientation du robot, et un Lidar LMS291-S05 pour obtenir les distances. Les bibliothèques pour les servo-commandes et pour l'IMU sont spécifiquement écrites sous ROS, alors qu'une bibliothèque générale pour le Lidar, développée par la communauté ROS, a ete utilisée. Le résultat est une plateforme opérationnelle qui participe à la recherche dans le domaine des véhicules terrestres autonomes ; par exemple en réalisant des expériences pour étudier l'efficience et l'efficacité des nouveaux algorithmes pour traiter le problème de Localisation et Cartographie Simultanées (SLAM : Simultaneous Localization and Mapping) ainsi-nommé.

Mots-clés : Les Robots Mobiles Autonomes ; AGVs, ROS.

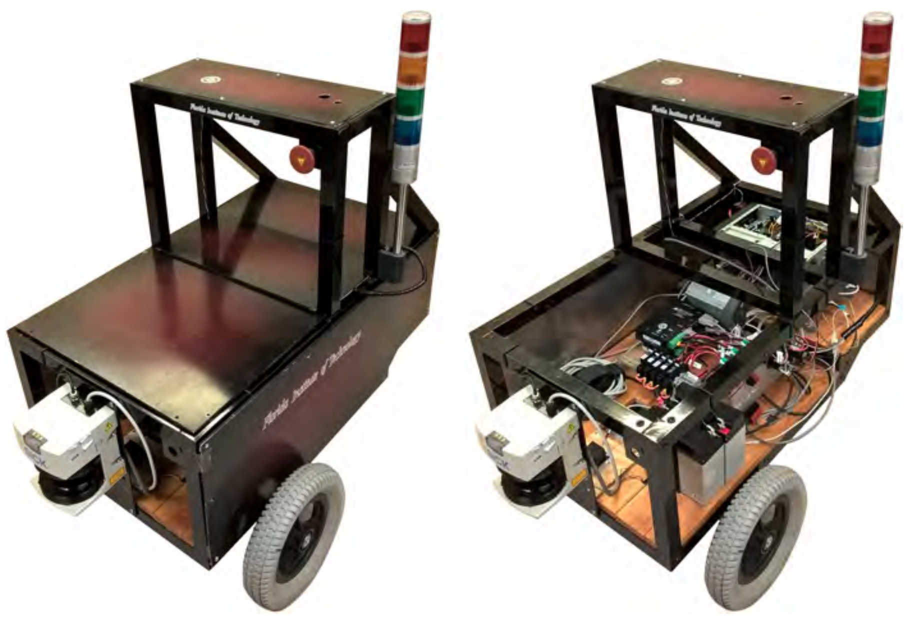
NOMENCLATURE

l	distance between two front wheels (m)
R	instantaneous turning radius (m)
r	radii of wheels (m)
u	rotational speeds of DC motors (RPM)
v	linear velocities (m/s)
Acronym	
CRC	cyclic redundancy check
ICC	instantaneous center of curvature
IGV	intelligent ground vehicle
IMU	inertial measurement unit
LiDAR	light detection and ranging
LSB	least significant bit
PID	proportional-integral-derivative
ROS	robot operating system
RPM	revolutions per minute
SOF	start of frame
Greek Symbols	
ω	angular velocities (rad/s)
Subscripts	
G	axis or origin of global frame
L	parameter of left wheel
R	parameter of right wheel
R	axis or origin of robot frame when in O_R, X_R, Y_R and Z_R

1. INTRODUCTION

Mobile robots have a history of more than 70 years, with the V-1 flying bombs during World War II as early examples of robotic weapons [1] [2]. But the earliest autonomous mobile robots are probably the "tortoises" Elmer and Elsie built by W. Grey Walter between 1948 and 1949 [3]. After decades of development, mobile robots have a wide variety in their architectures and functionalities nowadays.

As for this project, a differential drive three-wheeled vehicle (Figure 1) is chosen to develop a platform for further research on IGV, a mobile robot that aims at solving outdoor SLAM problem. It is designed to operate in urban environments such as concrete paths and artificial grass fields. The advantage of a differential drive wheeled system is that it is adequate for locomotion in such terrains and compare to other locomotion methods it is easy to be controlled, therefore more computational resources can be distributed to high level tasks such as localizing, map building, and other tasks that based on localization and mapping. In summary, this platform is designed for studying current techniques or developed new techniques of wheeled mobile robots.



(a) External View (b) Internal View

Fig. 1. The IGV

2. MECHANICAL SYSTEM

The chassis of this mobile robot was fabricated with Aluminum 6063-T5 tubings by the previous IGVC team. The dimensions of the chassis were measured manually during this project, therefore each measurement has a tolerance of $\pm 5\ mm$ (Figure 2(a)). Additionally, enclosures are applied to chassis to protect the internal electronic devices from direct sunshine or light rain when the robot goes outdoors (Figure 2(b), 2(c)).

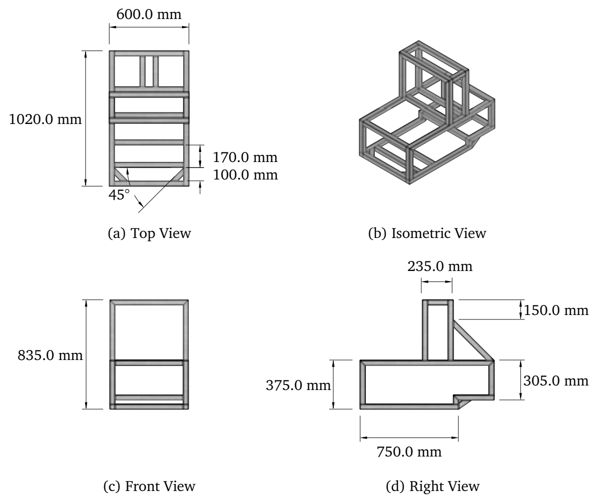
Three wheels with rubber tires are mounted under the chassis of IGV. The two front wheels are powered by two DC motors and the rear castor wheel is free to rotate (Figure 3(a)). To operate in a two dimensional plane, this differential drive vehicle uses only x and z component to represent its linear velocity and angular velocity by assuming no slipping occurs when the robot move forward/backward along its longitudinal axis x and rotate around its vertical axis z. As shown in Figure 3(b), (X_G, Y_G) is the global frame and (X_R, Y_R) is the robot frame with the origin point O_R at the midpoint of the two powered wheels. Linear velocity of the vehicle is defined at O_R (Figure 4(a)), which is given by Equation 1.

$$v = \frac{v_L + v_R}{2}\ (m/s) \tag{1}$$

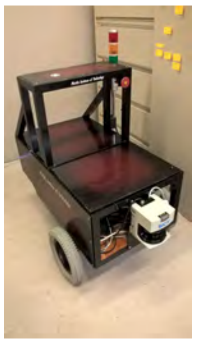
Velocities of the left and right front wheels are converted from the rotational speeds $u_{L,R}$ of the motors measured by RPM. The conversion is given by Equation 2.

$$v_{L,R} = \frac{u_{L,R}}{60} \cdot 2\pi r_{L,R}\ (m/s) \tag{2}$$

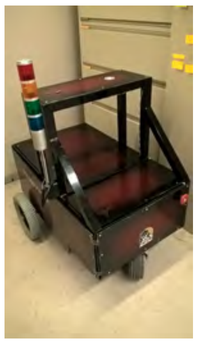
Where r_L and r_R are radii of the two front powered wheels. Due to the weight of robot itself, the actual measurement of the distance from the center of each motor shaft to ground is about



(a) Dimensions

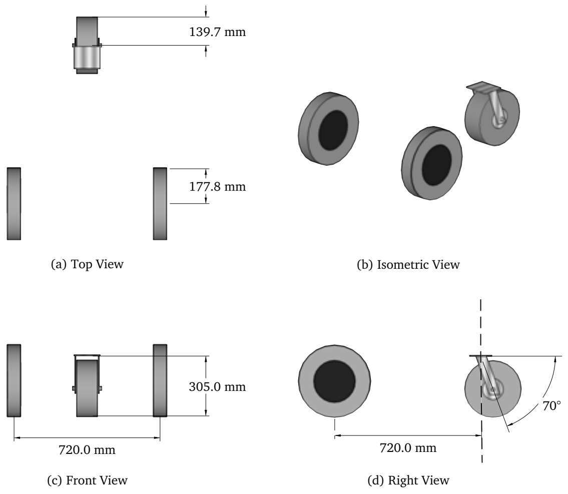


(b) Front View

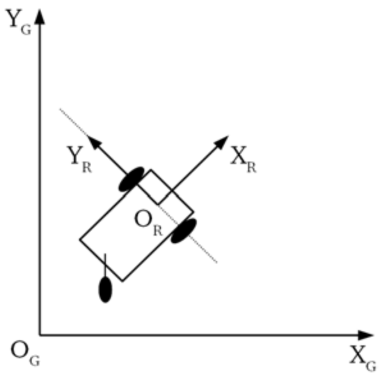


(c) Rear View

Fig. 2. Chassis Dimensions and Enclosures

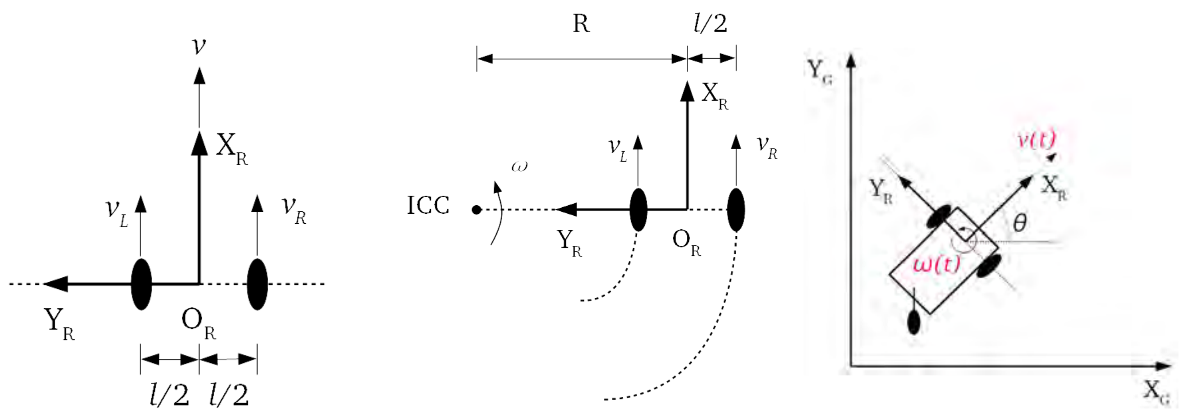


(a) Wheel Structure



(b) Global Reference Frame

Fig. 3. Wheel Structure and Global Reference Frame



(a) Linear Velocity (b) Angular Velocity (c) Twist Velocity

Fig. 4. Vehicle Velocities

160 *mm*, which makes the robot moves forward/backward $2 \cdot \pi \cdot r \approx 1005.3$ *mm* or about 1.005 *m* per revolution. Therefore the linear velocity of each wheel is given by Equation 3.

$$v_{L,R} = \frac{u_{L,R}}{60} \cdot 1.005 = \frac{67 u_{L,R}}{4000}\ (m/s) \tag{3}$$

According to Figure 4(b), the relationships between the angular velocity ω and linear velocities are given by Equation 4 [4].

$$\omega(R \mp \frac{l}{2}) = v_{L,R} \tag{4}$$

Solving for R and ω results in Equation 5.

$$R = \frac{l(v_L + v_R)}{2(v_R - v_L)}, \omega = \frac{v_R - v_L}{l} \tag{5}$$

Combining Equation **??**, 5, and $l = 0.72$ *m* given by Figure 3(a) results in Equation 6.

$$R = \frac{9(u_L + u_R)}{25(u_R - u_L)}, \omega = \frac{67(u_R - u_L)}{2880} \tag{6}$$

ROS expresses velocity with "Twist" message type [5] which breaks into linear and angular parts (Table 1). As a differential drive robot in a 2D map, only the x component in linear part and z component in angular part are nonzero and assigned respectively by v and ω defined above (Figure 4(c)). In order to control the robot, left and right motor rotational speeds need to be solved from v and ω as shown in Equation 7. The entire operation can be expressed by the following pseudocode.

Linear	Angular
float64 x	float64 x
float64 y	float64 y
float64 z	float64 z

Table 1. Twist Message Components

$$u_{L,R} = \frac{4000v \mp 1440\omega}{67} \tag{7}$$

```
Twist.linear.x = v;
Twist.angular.z = ω;
u_L=(4000*Twist.linear.x-1440*Twist.angular.z)/67;
u_R=(4000*Twist.linear.x+1440*Twist.angular.z)/67;
IGV.setVelocity_Left(u_L);
IGV.setVelocity_Right(u_R);
```

3. ELECTRONIC SYSTEM

3.1. Motion System

The two front wheels of IGV are driven by DC motors, of which the parameter specifications are shown in Table 2. Two DPRALTE-060B080 servo drives (Figure 5, Table 3, courtesy of [6]) are used to control the DC motors, which have built-in PID controllers and the gains applied to both are $K_P = 0.05$, $K_I = 0.2$, $K_D = 2.4 \times 10^7$. The velocity control performance is shown by Figure 6.

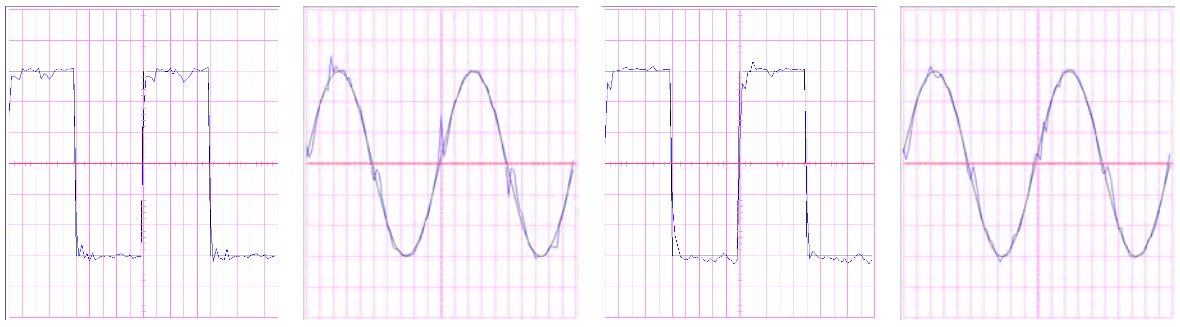
Parameter	Specification
Voltage	24 V
Peak Current	6.6 A
Continuous Current	3.3 A
Max Speed	110 RPM

Table 2. DC Motor Parameter Specifications

Fig. 5. AMC DPRALTE-060B080

Parameter	Specification
Peak Current	60 A
Continuous Current	30 A
Supply Voltage	20 – 80 VDC

Table 3. DPRALTE-060B080 Parameter Specifications



(a) Square (Left) (b) Sinusoidal (Left) (c) Square (Right) (d) Sinusoidal (Right)

Fig. 6. Left and Right Motors Velocity Control Responses

These servo drives are configured to be controlled by RS-485 serial communication. The serial command structure is defined by the manufacturer as shown in Figure 7. Using this command structure, the PC (master/host) can send a message to the servo drive (slave), for example setting the target velocity; or request information from the servo drive, such as reading the bridge status or motor odometry fed by encoder.

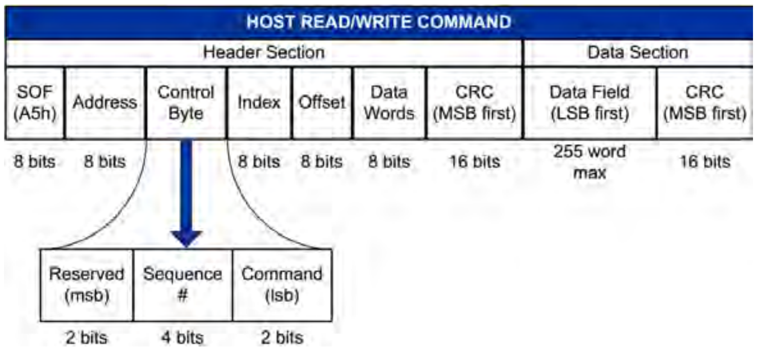


Fig. 7. Serial Command Structure

1. **SOF** Start byte of each message between a master and a slave, always $A5h$.
2. **Address** Message destination address. By default, the address is $3Fh$ and it was assigned to the right servo drive , while $3Eh$ was assigned to the left servo drive.
3. **Control Byte** For this project, only $01h$ (request information) and $02h$ (write command) have been used in this byte. And for velocity control, only $02h$ is used to send target from PC to servo drives. Although $01h$ doesn't affect the "write" operation at all, it is still necessary. The importance of it will be explained later.
4. **Index** Always $45h$, represents using serial communication.
5. **Offset** Used with respect to "Index" to specify the communication channel. The values are $00h$ and $02h$ for left and right respectively.
6. **Data Words** The number of words (2 bytes) in "Data Field". The data (velocity) for serial communication has to be a $32-bit$ (2 words) integer, therefore this byte is always set to $02h$.
7. **CRC (header section)** The CRC here is referred to CRC-16-CCITT (Xmodem) and based on the polynomial $X^{16}+X^{12}+X^5+1$.
8. **Data Field** In this project, it should be a hexadecimal target velocity measured by "counts/sec" demanded by the encoders. Using 10 RPM as an example, the conversion from RPM to "counts/sec" is given by Equation 8 and it will be filled into the field in LSB first order.

$$10\ rev/min * 400\ counts/rev * 1\ min/60\ sec * \frac{2^{17}}{20000\ Hz} \approx 437 = 000001B5h \tag{8}$$

9. **CRC (data section)** Computed based on the value of data field.

To sum up, a complete serial command that sets target velocity would be as shown in Figure 8 (using address $3Fh$ and target velocity 10 RPM as an example). The gray colored header section can be preset since it will not change during the entire operation.

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

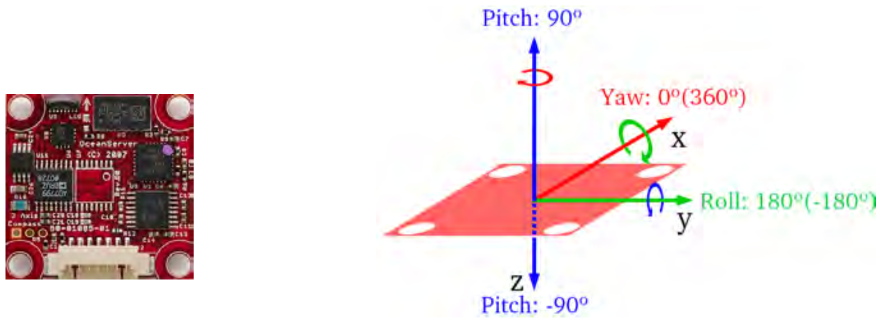
Fig. 8. Serial Command Example

3.2. Orientation Sensing System

A 3-axis OS5000-S IMU (Figure 9(a), courtesy of [7]) is used to read the yaw, pitch and roll angles. The output format of the angle message is:

$$\$Cyyy.yPpp.pRrrr.r* < checksum >$$

Where "yyy.y" is yaw angle from 0° to 360°, "pp.p" represents pitch angle from $-90°$ to 90°, "rrr.r" stands for roll angle from $-180°$ to 180° (Figure 9(b)). The keywords "C", "P" and "R" can be used as delimiters to extract information from the raw data by using the "std::string::find" function [8].



(a) OS5000-S (b) Inclination Range

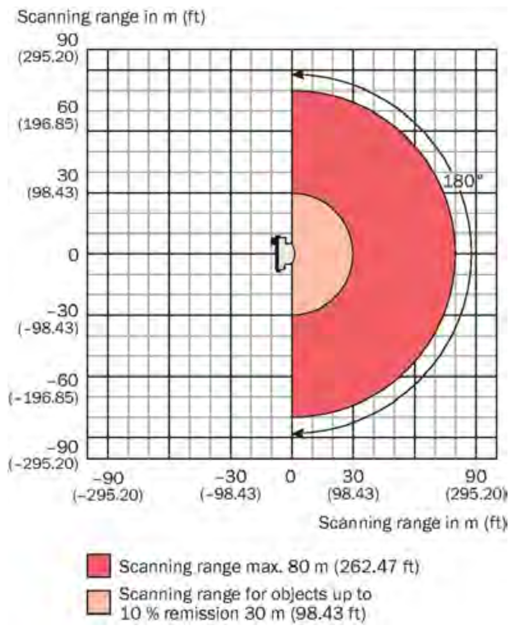
Fig. 9. OS5000-S IMU

3.3. Ranging System

A LMS211-S07 LiDAR (Figure 10(a), courtesy of [9]) is mounted on front of the vehicle. It has a scanning range up to 80 m and a range of 30 m for up to 10% object remission (Figure 10(b), courtesy of [9]). The Sick LiDAR Matlab/C++ toolbox [10] and its ROS wrapper [11] provides a visualization of the readings. An example can be accomplished by the following command lines [12], of which the result is shown by Figure 11(b).

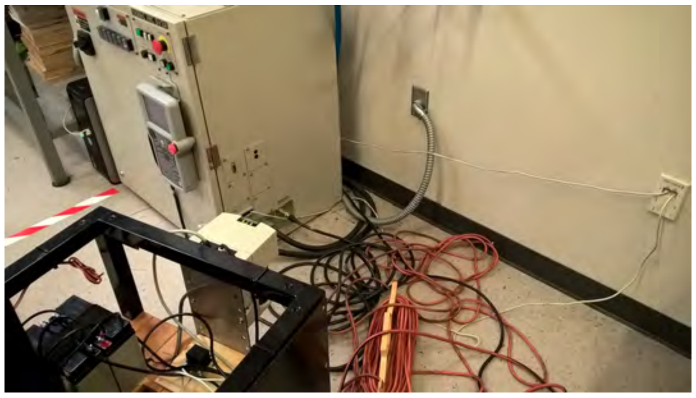
```
sudo chmod a+rw /dev/ttyUSB0
roscore
rosrun sicktoolbox_wrapper sicklms _port:=/dev/ttyUSB0 _baud:=9600
rosrun rviz rviz
```

(a) LMS291-S05

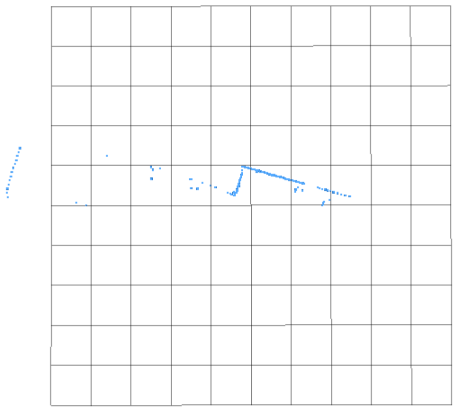


(b) Operating Range

Fig. 10. LMS291-S05 LiDAR



(a) LiDAR Facing a Corner



(b) Visualization by ROS's RViz

Fig. 11. LiDAR Data Visualization

4. SOFTWARE SYSTEM

The ROS manages the operation of every architecture of the IGV software system, which not only allows them to process tasks individually but also sets up a communication network to allow them to work cooperatively. ROS has three levels of concepts: filesystem, computation graph, and community level. The local PC hosts the first two levels and provides source codes to hardware drivers, while the community level will provide development assistance via Internet. A working example based on the current project is shown in Figure 12.

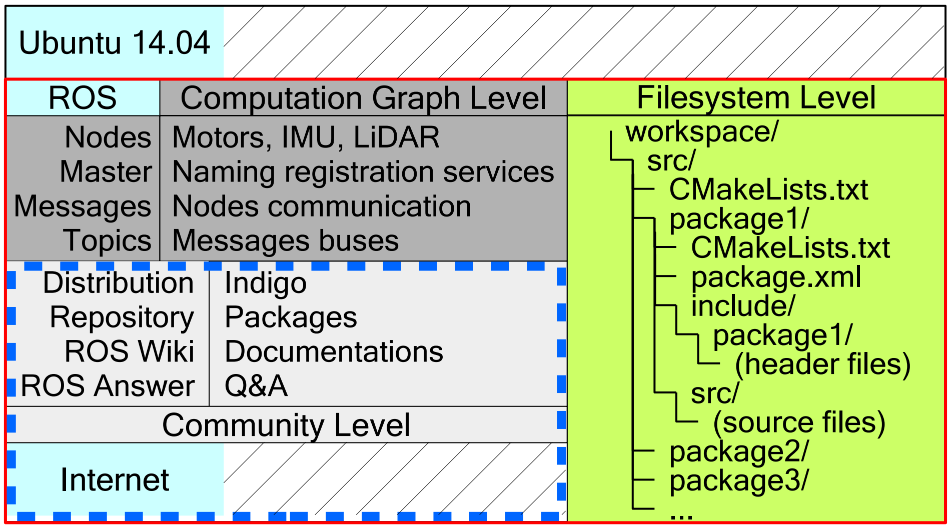


Fig. 12. A Working Setup of ROS

To make the PC exchange information with hardwares, the “serial” library [13] is used to setup serial communications. The challenging part for new developers would be making two packages work together. The following example shows the configurations in a CMakeLists.txt file to get use of the “serial” package.

```
cmake_minimum_required(VERSION 2.8.3)
project(dpralte060b080)
find_package(catkin REQUIRED COMPONENTS
    message_generation              # required to generate message files
    roscpp                          # package written in C++
    serial                          # used to communicate with PC
    std_msgs                        # message data type
)
add_message_files(FILES
    DPRALTE060B080_Msg.msg          # declares message files, which are
                                    # usually in /msg directory
)
generate_messages(DEPENDENCIES
    std_msgs
)
catkin_package(
    INCLUDE_DIRS include
    LIBRARIES dpralte060b080
    CATKIN_DEPENDS message_runtime roscpp serial std_msgs
    DEPENDS system_lib
)
include_directories(include
    ${catkin_INCLUDE_DIRS}
)
add_library(dpralte060b080
    src/DPRALTE060B080.cpp
)
add_executable(igv_DPRALTE060B080
    node/DPRALTE060B080_Node.cpp # node file, just like "main.cpp"
                                 # generated by many IDEs
                                 # (e.g. CodeBlocks)
    src/DPRALTE060B080.cpp
)
add_dependencies(igv_DPRALTE060B080
    ${${PROJECT_NAME}_EXPORTED_TARGETS}
    ${catkin_EXPORTED_TARGETS}
)
target_link_libraries(igv_DPRALTE060B080
    ${catkin_LIBRARIES}
)
```

5. CONCLUSION

The design and development of a mobile platform to support autonomous ground vehicle research has been presented. The mobile platform's kinematics were analyzed, a servo control system was implemented, and sensors including an IMU and LiDAR have been incorporated into the system. The APIs of the related ROS packages are provided [14]. This which enables other researchers to quickly implement algorithms that address the so called Simultaneous Localization and Mapping (i.e. SLAM) problem and perform experiments. The result is a working platform that is now supporting active research in autonomous ground vehicles. Ongoing development work is focused on adding additional sensors (e.g. gps & stereo vision) and related ROS APIs to the system.

REFERENCES

1. Vanek, D. *Fulfilment: Memoirs of a Criminal Court Judge*. Dundurn. ISBN 155002325X, 1999.
2. Krishnan, A. *Killer Robots: Legality and Ethicality of Autonomous Weapons*. Routledge. ISBN 0754677265, 2009.
3. Holland, O. "Grey Walter: The Pioneer of Real Artificial Life." In "Artificial Life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems," pp. 34 – 44. MIT Press, Nara, Japan. ISBN 0262621118, May 1997.
4. Dudek, G. and Jenkin, M. *Computational Principles of Mobile Robotics*. Cambridge University Press.

ISBN 0521560217, 2000.

5. ROS. "Twist Message." `http://docs.ros.org/api/geometry_msgs/html/msg/Twist.html`. Accessed: 2017-01-29, January 2017.
6. Advanced Motion Controls. "DPRALTE-060B080." `http://www.a-m-c.com/download/datasheet/dpralte-060b080.pdf`. Accessed: 2017-01-22, April 2016.
7. Technology, O. "Digital Compass Users Guide, OS5000 Series." `http://www.ocean-server.com/download/OS5000_Compass_Manual.pdf`. Accessed: 2017-01-23, 2015.
8. cplusplus.com. "std::string::find." `http://www.cplusplus.com/reference/string/string/find/`. Accessed: 2017-01-25.
9. Sick. "LMS211-S07." `https://sick-virginia.data.continum.net/media/pdf/3/53/653/dataSheet_LMS211-S07_1018966_en.pdf`. Accessed: 2016-01-23, August 2016.
10. Hendrix, J. "The Sick LIDAR Matlab/C++ Toolbox." `http://sicktoolbox.sourceforge.net/`. Accessed: 2017-01-23, November 2010.
11. Rockey, C. "sicktoolbox_wrapper." `http://wiki.ros.org/sicktoolbox_wrapper`. Accessed: 2017-01-23, October 2013.
12. Clbaughe. "Using SICK Laser Scanners with the sicktoolbox_wrapper." `http://wiki.ros.org/sicktoolbox_wrapper/Tutorials/UsingTheSicklms`. Accessed: 2017-01-23, February 2013.
13. Hendrix, A. "serial." `http://wiki.ros.org/serial`. Accessed: 2017-01-25, February 2014.
14. Li, J. *A Robot Platform for Intelligent Ground Vehicle Research*. Master's thesis, Florida Institute of Technology. Accessed: 2017-01-29, November 2016.