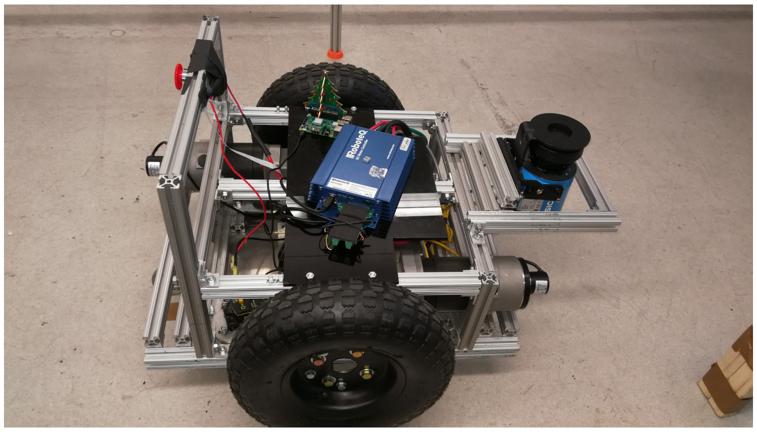
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
THE UNIVERSITY OF TEXAS AT ARLINGTON

SYSTEM REQUIREMENTS SPECIFICATION
SENIOR DESIGN
SUMMER/FALL 2017



TEAM UGV
UGV

CHRIS COLLANDER
DARRELL RASCO
CHASE HUFFMAN
PAUL ASYN

Revision History

Revision	Date	Author(s)	Description
0.1	07.20.2017	Darrell Rasco	document creation and initial editing
1.0	12.07.2017	Chris Collander	finalized document

Contents

List of Figures

1 Product Concept

1.1 Purpose and Use

The system consists of an Unmanned Ground Vehicle (UGV) communicating through WiFi with ROS-Kinetic to a centralized desktop PC running a Linux distribution. The PC will send motor messages to the UGV for navigation and read in information from a spinning LiDAR on the UGV to interpret the environment. After interpreting the environment and receiving a command of where to go in the environment, the PC will continuously send motor messages to the UGV navigation to the destination.

The primary goal of this project is to create a payload-carrying UGV that can be controlled through ROS to allow human robot collaboration for tasks involving a number of repetitive, precise, and/or intellectual steps.

1.2 Intended Audience

This product is primarily designed for researches in goals involving human robot collaboration.

2 Product Description

2.1 External Inputs & Outputs

Inputs will be from the user on the PC software and the LiDAR data from the UGV. The outputs will be only the motors and brakes on the UGV.

2.2 Product Interfaces

The PC consists of a windowed software package displaying the current environment. The user will point and click various locations and the UGV will navigate to that location in the environment.

3 Customer Requirements

3.1 ROS

3.1.1 Description

The system shall use ROS-Kinetic for settings and controls.

3.1.2 Source

Chris Collander - Team member

3.1.3 Constraints

N/A

3.1.4 Standards

N/A

3.1.5 Priority

High

3.2 LiDAR Usage

3.2.1 Description

The system shall use a LIDAR sensor to map its environment. A LIDAR sensor placed on the vehicle will publish data to the /scan ROS topic.

3.2.2 Source

Chase Huffman - Team member

3.2.3 Constraints

N/A

3.2.4 Standards

N/A

3.2.5 Priority

High

3.3 Hector Slam

3.3.1 Description

The system shall use the Hector Slam algorithm for environment recognition which shall be implemented through ROS subscribing to the /scan topic from the LiDAR.

3.3.2 Source

Chase Huffman - Team member

3.3.3 Constraints

N/A

3.3.4 Standards

N/A

3.3.5 Priority

High

3.4 Raspberry Pi

3.4.1 Description

The system shall use a Raspberry Pi onboard the UGV.

3.4.2 Source

Paul Asyn - Team member

3.4.3 Constraints

N/A

3.4.4 Standards

N/A

3.4.5 Priority

High

3.5 Raspberry Pi ROS Handling

3.5.1 Description

The Raspberry Pi on the UGV shall subscribe to a topic published by the PC to receive motor values in the form of a two-bit binary value for brake controls and a pair of unsigned integers for rotational velocity values.

3.5.2 Source

Paul Asyn - Team member

3.5.3 Constraints

N/A

3.5.4 Standards

N/A

3.5.5 Priority

High

3.6 Python Programming Language

3.6.1 Description

The software for the system shall be written exclusively in Python 2.7.

3.6.2 Source

Chase Huffman - Team member

3.6.3 Constraints

N/A

3.6.4 Standards

N/A

3.6.5 Priority

High

3.7 Systems

3.7.1 Description

The system shall consist of an external Linux Desktop PC controlling the UGV.

3.7.2 Source

Chris Collander - Team member

3.7.3 Constraints

N/A

3.7.4 Standards

N/A

3.7.5 Priority

High

3.8 System Communication

3.8.1 Description

The PC and the UGV shall communicate exclusively through ROS. Other ROS enabled devices may be controlled in the same software.

3.8.2 Source

Chris Collander - Team member

3.8.3 Constraints

N/A

3.8.4 Standards

N/A

3.8.5 Priority

High

3.9 Omni-Directional Wheels

3.9.1 Description

The UGV shall use at least two Omni-Directional wheels for movement.

3.9.2 Source

Paul Asyn - Team member

3.9.3 Constraints

N/A

3.9.4 Standards

N/A

3.9.5 Priority

High

4 Performance Requirements

4.1 Navigation

4.1.1 Description

The UGV shall be able to navigate to and from locations without disturbing the payload.

4.1.2 Source

Darrell Rasco - Team Member

4.1.3 Constraints

N/A

4.1.4 Standards

N/A

4.1.5 Priority

High

4.2 Obstacles

4.2.1 Description

The UGV shall attempt to change course if an obstacle is detected. If no course change is possible, then the UGV shall alert the user.

4.2.2 Source

Darrell Rasco - Team Member

4.2.3 Constraints

N/A

4.2.4 Standards

N/A

4.2.5 Priority

High

4.3 Environment

4.3.1 Description

The UGV shall be able to easily move on a flat floored environment such as the senior design lab

4.3.2 Source

Chris Collander - Team Member

4.3.3 Constraints

N/A

4.3.4 Standards

N/A

4.3.5 Priority

High

4.4 Environment

4.4.1 Description

The UGV shall be able to easily move on a flat floored environment such as the senior design lab.

4.4.2 Source

Chris Collander - Team Member

4.4.3 Constraints

N/A

4.4.4 Standards

N/A

4.4.5 Priority

High

4.5 Speed

4.5.1 Description

The UGV shall be able to reach speeds higher than 5 miles per hour.

4.5.2 Source

Chris Collander - Team Member

4.5.3 Constraints

N/A

4.5.4 Standards

N/A

4.5.5 Priority

High

4.6 Payload Weight

4.6.1 Description

The UGV shall be able to carry a payload weight of greater than 10 pounds.

4.6.2 Source

Chris Collander - Team Member

4.6.3 Constraints

N/A

4.6.4 Standards

N/A

4.6.5 Priority

High

4.7 Running Time

4.7.1 Description

The UGV shall be able to run continuously for at least 30 minutes.

4.7.2 Source

Chris Collander - Team Member

4.7.3 Constraints

N/A

4.7.4 Standards

N/A

4.7.5 Priority

High

5 Safety Requirements

5.1 Local Fused Shutoff

5.1.1 Description

Upon pulling too much current, the UGV shall trip a Fuse or Circuit Breaker to open circuit the power to every component of the UGV.

5.1.2 Source

Chris Collander - Team Member

5.1.3 Constraints

to be determined

5.1.4 Standards

N/A

5.1.5 Priority

Critical

5.2 Remote Shutoff

5.2.1 Description

Upon receiving a specific ROS message, the UGV should immediately power off, either through software or by tripping a circuit breaker.

5.2.2 Source

Chris Collander - Team Member

5.2.3 Constraints

N/A

5.2.4 Standards

N/A

5.2.5 Priority

Critical

5.3 Packet Loss

5.3.1 Description

The UGV shall lock all breaks when a ROS motor message has not been received within two seconds.

5.3.2 Source

Chris Collander - Team Member

5.3.3 Constraints

N/A

5.3.4 Standards

N/A

5.3.5 Priority

Critical

References