Feb 24, 2013

Teams 19 & 20: Communications Protocol Specification Document

Laboratory #2: Requirements and Specification

Morgan, Laura
Miaw, Jireh
Hauser, Steven
Dworak, Catherine
Bertoglio, David
Lenig, Tyler
Tang, Raymond
Rupakhetee, Archit
McMillion, Andrew

Work Product

This document describes the communication protocol implemented by Teams 19 and 20 for communication between the base station control system, and the robot. This document describes the creation, and decoding process for messages.

Document Revision Information

2/15/2013 created
2/17/2013 designed base station to robot messages
2/22/2013 continued design
2/24/2013 designed robot to base station messages
3/17/2013 added commands and error detection

Approval Sheet

All group members whose names are listed below approve of the document and contributed fairly.

Member Names

Group 19 representative

Dworak, Catherine

Group 20 representative

Lenig, Tyler

Pledge

On my honor, as a student, I have neither given nor received unauthorized aid on this assignment.

Names

Group 19

Morgan, Laura

Miaw, Jireh

Hauser, Steven

Dworak, Catherine

Bertoglio, David

Group 20

Lenig, Tyler

Tang, Raymond

Rupakhetee, Archit

McMillion, Andrew

Table of Contents

Introduction

About

This document describes the protocol used to communicate between robot and base station system. This protocol allows the base station to control the robot, and allows the robot to send messages including errors to the base station.

Protocol Description

This protocol uses 11-character messages to communicate between the robot and the base station. The first 10-character of messages encode both commands from the base station to the robot and messages from the robot to the base station. The messages are structured such that the first two characters determine the type of command or message. The remaining characters are used for various parameters that are documented below. The 11th character holds the checksum used for error detection.

Base Station to Robot Messages

Command Structure

Commands are 10-character messages, where the first two characters are the command type. The remaining characters represent parameters to the command, used by the robot to determine how to execute the command.

No-Op

Message: 0000000000
Description: This command is the no operation command can be used to test if messages are being sent. This message is a "null" message.

Move Straight

Command Type: MS
Parameters: Forward/Backwards, and distance.

Character 2 is forward or backwards (F/B)
Characters 3-9 are distance (#), can be null (0s)

Description: This command moves the robot in a straight line. The forward/backward parameter control the direction the robot will move in. The distance allows for the robot to move a specified distance, this parameter can be null. If distance is null, the robot will continually move
Example Commands:

MSF0000000 will move the robot forward continuously.
MSB0001000 will move the robot backwards 1000 units.

Move Arc

Command Type: MA
Parameters: Forward/Backwards, left/right, radius, distance

Character 2 is forward or backwards (F/B)
Character 3 is left or right (L/R)
Characters 4-6 are radius (# degrees)
Characters 7-9 are distance (#), can be null (0s)

Description: This command moves the robot in an arc. The forward/backward parameter control the direction the robot will move along the arc. Left/Right will control the direction the robot arcs to. Radius is the absolute value of the number of degrees to move. The distance allows for the robot to move a specified distance, this parameter can be null. If distance is null, the robot will continually move until stopped.
Example Commands:

MAFL090000 will move the robot forward to the left along a 90 degree curve continuously

MABR030100 will move the robot backwards along a 30 degree curve for 100 units.

Turn

Command Type: TN

Parameters: Left/Right, and radius

Character 2 is left or right (L/R)

Characters 3-9 are radius (# degrees), can be null (0s)

Description: This command turns the robot when stationary. The Left/Right parameter determines the direction the robot turns. The Radius parameter is an absolute value that determines how far the robot turns. If the radius is null, the robot continually turns until stopped.

Example Commands:

TNR0000090 will turn the robot right 90 degrees

TNL0000000 will turn the robot left continuously

Stop

Message: ST00000000

Description: This command stops any actions that the robot is currently doing. This will end any movement actions.

Read Sensor

Command Type: RS

Parameters: Sensor Port

Character 2 is sensor type (U for Ultrasonic, T for touch, M for sound, L for light)

Characters 3-9 are 0

Description: This command will read a specified sensor. The Sensor Port parameter will determine which sensor to read the value of.

Example Commands:

RSU0000000 will cause the robot to read the value of the sensor, and send the data to the base station.

Set Speed

Command Type: SS

Parameters: Motor/Motor Combination, and new speed.

Character 2 is Motor/Motor combination (A for Motor A, B for Motor B, C for Motor C, D for Drive Motors)

Characters 3-9 are the new speed

Description: This command will change the speed of the motors. The combination

will determine which motors or combinations of motors to change the speed for.
Example Commands:

Read All Sensors

Command Message: RA00000000
Description: This command tells the robot to read all sensors and send the data. Each sensor's data will be sent to the base station in a separate message.

End Connection

Command Message: EC0000000
Description: This command instructs the robot to end connection with the base station.

Robot to Base Station Messages

Acknowledgment

Description: This message is sent to the base station as acknowledgment of receiving a command.
Message: AK00000000

Error Messages

Sensor Error Messages

Message Type: ERS
Parameters: Message number
Characters 3-9 are message number
Description: This message will tell the base station that an error with a sensor has occurred. The message number maps to a more specific description, that the base station will have stored locally for reference. Available messages can be seen in a table below, which will have additions added as required. Errors for sensors is only if the sensor is disconnected. For the bluetooth sensor, it is if the connection is disconnected from the base station.

Message Number	Description
0000001	Error with sensor in port 1
0000002	Error with sensor in port 2
0000003	Error with sensor in port 3
0000004	Error with sensor in port 4

Motor Error Messages

Message Type: ERM

Parameters: Message Number

Characters 3-9 are message number

Description: These messages will tell the base station that an error with a motor has occurred. The message number correlates to a specific description, which the base station has stored locally. Available messages can be seen in a table below, which will have additions added as required. Errors for the motor includes if the motors are not connected, if the speeds are faster than set limit, and if the motors are stuck.

Message Number	Description
0000001	Error with motor in port A
0000002	Error with motor in port B
0000003	Error with motor in port C

Sensor Data Messages

Message Type: SD

Parameters: Sensor Type, and Data

Character 2 is sensor type (U for Ultrasonic, T for Touch, M for Sound, or L for Light)

Characters 3-9 sensor data

Description: These messages allow for the robot to send data to the base station based on the values of the sensor.

Error Detection

Introduction

The Error detection this protocol utilizes is a checksum for detecting errors in packets, and a timeout on acknowledgments. The checksum is calculated using only the first 10 characters of the message, then checked against the character that is sent in the packet. The timeouts will be 10 seconds before the sender will assume the packet was lost and needs to be retransmitted.

Checksum Function

The function for calculating the checksum is

$(checksum = \sum_{i}^{10}(byte)message[i])\ mod\ 256$. This function allows for no matter the value of the checksum it will fit in a one-byte character. The function is the sum of the byte value of each character in the message modulo 256.