**RoboTrike**

**Tim Menninger**

**EE/CS 051**

**Fall 2014**

Description: The system is RoboTrike, a three-wheeled robotic car that will use holonomic motion to move. It will be controlled by a human with a keypad and serial port that connects a controller to a motor handler. In addition to motors to move the robot, it also has a laser. The controller has a screen with 14-segment digits on which the status can be displayed 8 characters at a time. There are 16 keys in a 4 x 4 grid that the user can press to make the robot move.

Global Variables: None.

Inputs: Serial Port – The controller uses the serial port as input to get strings to be displayed from the motor. These strings include motor statuses (speed, direction, laser status) and errors.

Serial Port – The motor target board uses the serial port as input to get strings corresponding to motor instructions. It then parses those characters/strings to determine what to do with its life.

Buttons – There are 16 buttons in a 4 x 4 square on both target boards. The buttons on the motor side target board ignores all 16 buttons. The buttons on the controller side target board correspond to the following actions, with each square a button and NW, N, NE, etc. the cardinal directions.

Figure 1: Button Events

|  |  |  |  |
| --- | --- | --- | --- |
| Fire Laser | Cease Fire |  | Speed Up |
| NW | N | NE | Slow Down |
| W | Stop | E |  |
| SW | S | SE |  |

Outputs: Serial Port – On the controller side, the serial port outputs commands to the motor side to relay to the motors/laser.

Serial Port – On the motor side, the serial port outputs motor statuses, such as direction, speed and laser status. It also outputs error descriptions. All of the outputs are strings that end with ASCII null character.

Display – 14 digit display on the controller side displays error descriptions should there be one and motor statuses.

Motors – There are three DC motors on the roboTrike that move the robot around via holonomic motion.

Laser – A laser can be fired or turned off.

User Interface: General:

For simplicity, we will name the buttons one through sixteen from top left to bottom right, counting upward first to the right then down. So the button second from the left and second from the bottom would be button number 6.

Movement:

Direction of movement uses buttons 5 through 7, 9, 11 and 13 through 15 (outlining a 3x3 square with its bottom left corner at the bottom left of the keypad). It sets an absolute angle of motion in the cardinal direction described in Figure 1. That is,

Move north Button 6

Move northeast Button 7

Move east Button 11

Move southeast Button 15

Move south Button 14

Move southwest Button 13

Move west Button 9

Move northwest Button 5

When a button is pressed, the display will show in what direction the trike is moving as “D” and then the angle counted positively clockwise from straight ahead.

Speed is controlled by the top two buttons of the rightmost row (buttons 4 and 8). Button 4 will cause the trike to speed up while button 8 will cause the trike to speed down. Trying to speed up while already going full speed will not change the speed. Similarly, trying to slow down past minimum speed will not change the speed of the trike. When a speed button is pressed, it steps once. If it is held, it will continuously step until released. When the speed is changed (or in the case of max and min speeds, attempted to be changed), the current speed is shown on the display of the controller side target board as “S” and then the speed with no space. Button 10, in the middle of all of the directional buttons, will reset the speed to zero, thereby stopping the Trike. The display will update accordingly.

Weapons:

The laser can be fired by pressing button 1. When the laser is fired, the display will read “Laser On”, taken from the motor side. This is a confirmation that the laser was successfully set, as opposed to a reiteration of the command the user sent. The laser can be turned off by pressing button 2. Upon turning the laser off, the display will read “Laser Off”, again from the motor side target board.

Display:

The display will use the 14-segment characters. If the message to be displayed is longer than 8 characters, it will show the first 8 characters for a second, then scroll by moving each character over to the left. When the end of the message is reached, it will continue displaying the last 8 characters (which may or may not be the entire string) until another string is to be displayed.

Error Handling: The system handles events by enqueuing their event codes. It is assumed that the system can dequeue much faster than it can enqueue, so if the event queue fills, then there has been a fatal error. In this case, the controller side display will show “FATAL ERROR”, and the board that experienced the issue will reset.

If there is an error as a result of serial communication, then the board that detected it will instruct the display on the controller side board to display a brief description of the error.

If there is an error within the finite state machine, which is used by the motor side target board to parse serial characters, then the display will read “PARSER ERROR”. This is not a fatal error and can be recovered from without rebooting.

Algorithms: The motor side target board uses a finite state machine to parse characters. This allows it to respond to a character read from the serial port with a knowledge of the preceding serial characters and anticipation of future ones.

Data Structures: Queues are used on both target boards to store events. They are enqueued as they arise (from interrupts, buttons, etc.) and dequeued as they are ready to be handled.

Limitations: There is no way to rotate RoboTrike. It cannot do it while moving laterally and it can also not rotate while moving, meaning it can only move in straight lines.

There are only 32K byes of RAM and 32K bytes of ROM available.

Known Bugs: None.

Special Notes: A good portion of the code in this has been adapted from or copied with permission from Glen George, and has been documented accordingly.