Exomars Rover Is My Robot

Daniel Estuardo González Sierra
Diego Ignacio Yon Díaz
Héctor Eduardo Hernández Ruiz
Óscar Rolando Ramírez Milián
Pablo Luis Pérez del Águila
Sandra Paola García Sandoval
Silvio Alejandro Urizar Salazar
Spaceapps Challenge, Guatemala, RoverChober

***Abstract*—In this paper, the details concerning to the development of the challenge *Exomars Rover Is My Robot*. Details concerning to the software and hardware are listed, including the PC interface, block diagrams and the finite state machines realized for the well-functioning of the rover. This paper also includes planned features to be added to this project in the future.**

I. Software

A. PC Software

In the start menu, the user selects over five options for the rover's movement. The first four options are for the user to manipulate the rover with the keyboard arrows. The difference between the moon and every planet is the reaction time that takes the rover to obey the order if it is in that location. The option five gives the rover the order to move by himself with the capacity of avoiding obstacles.

Figure 1. Interface main menu

When the user is manipulating the rover with the keyboard arrows, it has the option to move forwards, backwards and to cross either right or left. If the user selects to move forwards or backwards, he has to enter the amount of meters that he wants the rover to move; and to cross he has to enter de amount of degrees the rover has to turn. All of this information is sent to the rover via Wi-fi using a UDP protocol interface.

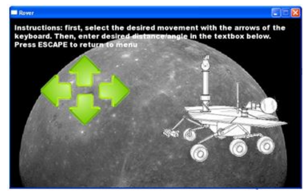

Figure 2. Interface moving option

B. Rover Software

The software for the rover was developed in C. The main structure of this software are two finite state machines, which govern the movement of the robot according to the inputs it recieves via the proximity sensor of the rover.
One of the machines controls the direction of the rover, and the other one is the one that controls the movement of the servomotor which points the proximity sensor in the desired direction and determines when the controller is able to read the sensor lectures. The state diagrams corresponding to both of the machines are presented below.

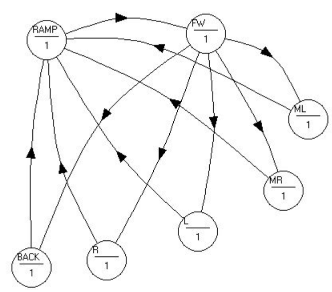

Figure 3. State diagram for the direction machine

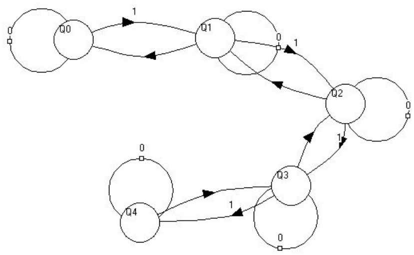

Figure 4. State diagram for the servomotor and sensing routines

The first machine, steering machine, consists of 7 states; which are: ramp, forward, mid-left turn, mid-right turn, left turn, right turn and backwards; which are denoted as Ramp, FW, ML, MR, L, R and BACK; respectively. The transition between states is determined by a 5-bit variable which is obtained via the servo state machine.
The second machine is designed to control the movement of the servomotor which points the proximity sensor in one of the 5 possible directions. Each state of this machine is a possible direction for the sensor. This directions are: $0^o, 45^o, 90^o, -45^o$ and -90^o, which are denoted in the diagram with $Q0, Q1, Q2, Q3$ and $Q4$; respectively.
The rest of the rover software are functions that make the state machines work. A PWM routine is included for the handling of the servomotor. Also, a dedicated function exists for controlling the H-Bridge for the steering of the motors and another one for capturing the value of the proximity sensor.

C. Interfaces

For communicating with the PC software. A wi-fi interface via a UDP protocol has been designed. This communication can be accomplished using a Raspberry Pi for acquiring the data that is sent via the PC-software.
The sensor-controller interface is a single-bit interface, so that the handling of the signals coming from the sensor can be easily handled by the controller. This saves the usage of ADCs programmed inside the controller.

II. Hardware

A. Controller

The micro-controller that we use, it's a tm4123gh6, mounted in a development board Stellaris Launchpad from Texas Instruments. We chose this development board due to its accesible price and programming flexibility. It can be programmed at low-level C language and possesses all the features needed to develop this completely.

B. Motors

For this project, two simple dc motors were used. This were powered using a 5V power source. These motors are controlled

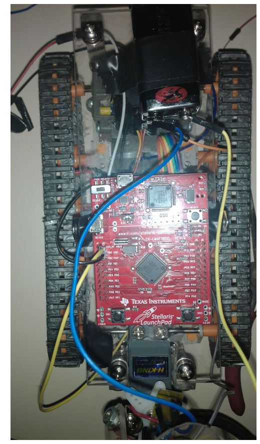

Figure 5. Stellaris board mounted in the project

using a H-Bridge board containing a LM-298 integrated circuit. The schematics for this integrated circuit is explained in the interfaces section.

C. Sensors

For the sensors, the rover contains an Allen Bradley 42KL-D1LB-A2 capacitive proximity sensor. This sensor works as a digital sensor. When it senses something at the specified distance, it delivers a logic 1 as an output. If it doesn't sense anything, it delivers a logic 0 in its output.

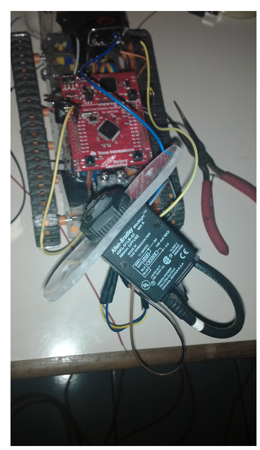

Figure 6. Proximity sensor mounted on the rover

D. Interfaces

For the controller to communicate with external circuitry, such as motors and the sensor, simple interfaces were implemented. For the motor movement, a H-bridge using a LM-298 integrated circuit was implemented. The schematics used for this board can be seen in the figure below.

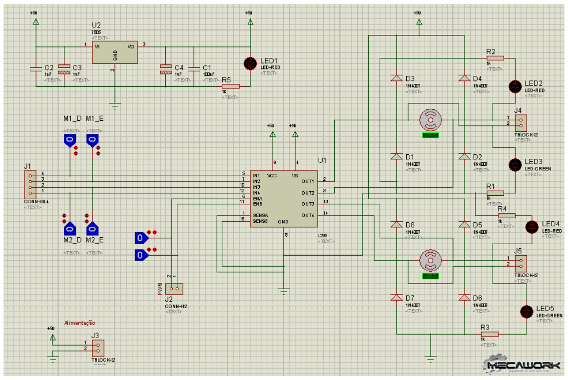

Figure 7. Schematics for the H-bridge board

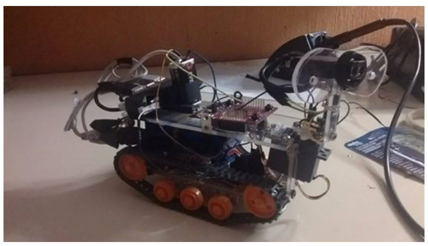

Figure 8. Rover’s final design

III. Future Development

Future plans for expanding the functions of the rover are on the making. More functions including extra software and hardware can be added to this rover.
First, the rover is ready to run all on batteries. The only thing needed is a battery strong and liable enough to guarantee that the rover is going to move for a considerable time.
Also, the PC interface has the option for the user to control the rover manually. This hasn’t been implemented completely. The UDP server is mounted in the Raspberry Pi and it can receive instructions. Further programming of the controller is required to implement the needed instructions for it to work as desired.
Another feature that has been planned is to implement a video streaming server using a Raspberry Pi. This can be easily implemented with it. Tests have been done and they have been succesful. Though, it hasn’t been implemented completely, yet.
In the hardware section. It is plausible to implement humidity and temperature sensors, so the rover can know better its environment and make more complex decisions depending in more variables. Also, a PID controller for the better-functioning of the rover’s motors can be implemented. For this, encoders can be adapted manually into 2 wheels of the motor and the PID to the controller.

IV. Project goals

The main goal of this project is mainly educational. This rover scale model can be used to teach elementary school and high school studends about technology and robotics in our country. With this rover and the experience of handling it, people can realize that technology isn’t that far away from third-world countries as ours as people usually think.
Also, kids can get interested on scientific careers such as physics and engineering. Projects like Exomars Rover is my Robot can develop interest in people so that they become scientists and can boost technology in Guatemala.