Journal of Multidisciplinary Engineering Science and Technology (JMEST)
ISSN: 2458-9403
Vol. 5 Issue 10, October - 2018

Remote Control via Android for a Small Vehicle's 2-Wheels Balancing

S.Melkonian
Dept. of Industrial Design and Production Eng.
University of West Attica
Athens, Greece

A.Chatzopoulos
Dept. of Industrial Design and Production Eng.
University of West Attica
Athens, Greece

M.Papoutsidakis
Dept. of Industrial Design and Production Eng.
University of West Attica
Athens, Greece

D.Piromalis
Dept. of Industrial Design and Production Eng.
University of West Attica
Athens, Greece

***Abstract*— The purpose of this thesis is the balancing of a vehicle on two wheels. The project is about a 4-wheeled, rear-wheel drive car, which will be able to get on its two back wheels and balance itself without any help from external factors. In addition, vehicle will be controlled wirelessly with the use of a cell phone. The robot’s balancing is possible by using a sensor, which is using a formula to read the tilt of the car with precision, and it passes it down to Arduino (microcontroller). This sensor is a combination of an accelerometer, gyroscope, compass and it is the most crucial part of the project, because without it the robot would never balance. This data goes through to the microcontroller, where it is processed and based on what it has received it moves the wheels accordingly. Obviously, that is not microcontroller’s only job, it also runs a PD controller that based on the tilt and angular velocity adjusts the power of the engines. The second part of the project is about the wireless communication between a cell phone and the vehicle. This is achieved with the help of a Wi-Fi module (receiver/transmitter). The module creates its own access point on which you can connect with a cell phone. Using an interface it is possible to send calls to the car, e.g. move forward, turn right etc. The vehicle has been put together entirely by me, because of the lack of similar projects every piece is purchased separately and it is not part of any kind of set. Many hours have been spent to achieve the best results, mainly held back by the asymmetrical distribution of weight.**

Keywords—Robot; Self-balanced; Arduino applications, platform control

I. Introduction

First, all the electrical components of this project will be mentioned as hyperlinks with their technical specifications:

- Arduino Uno Rev3
- 7A Dual Dc Motor Driver
- MPU-9250
- Wemos D1 R2 V2.1
- Powerex Rechargable Batteries
- DC Motors

TABLE I. DC MOTORS SPECIFICATIONS

Operation Range	6 – 28 V
Nominal Operation Voltage	12V
No Load Speed	500 rpm
No Load Current	≤ 200 mA
Rated Speed	380 rpm
Rated Current	≤ 670 mA
Rated Torque	1.2 kg*cm
Stall Current	2.19 A
D Output Shaft Diameter	6 mm
Weight	210 gr
Gearbox Diameter	37 mm
Motor Diameter	34 mm
Gearbox	10:1

List of all the components used in this project:

- Arduino Uno
- Wemos D1 R2 V2.1
- 7A Dual DC Motor Driver
- MPU-9250
- 2 DC Motors 12V 37mm
- 2 Motor mounts 37mm
- 10 AA Battery pack
- 10 AA Batteries
- 4 Wheels
- 4 Wheel adapter hex 12mm
- Precision shaft
- 2 Shaft supports
- 4 Ball bearings
- Wires male-to-male, male-to-female

- 2 Coaxial power connector 2.1mm
- Bolts M3/M4 with nuts
- Foam Layer (black)

The schematic of the project:

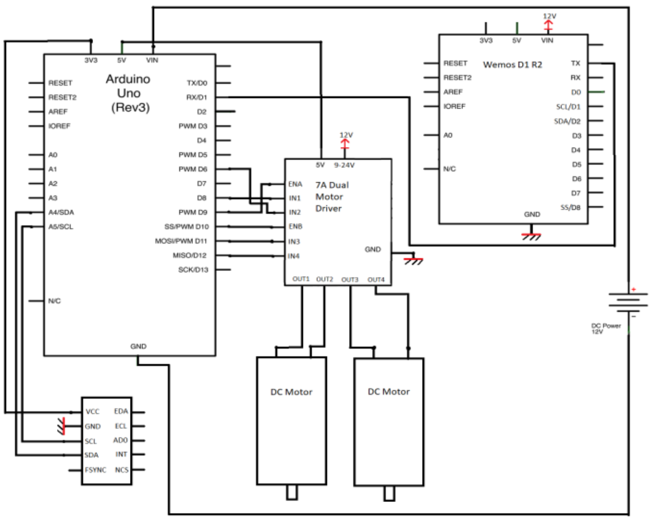

Fig. 1. Electrical Blueprint

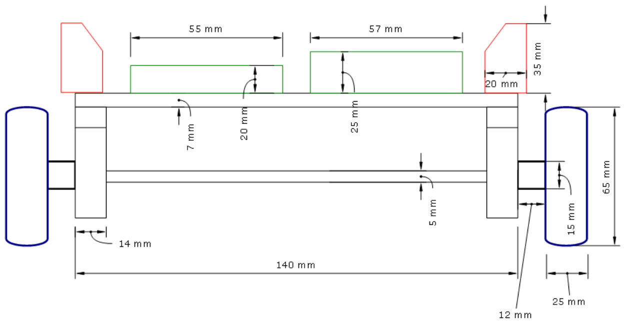

Fig. 2. Mechanical design front view

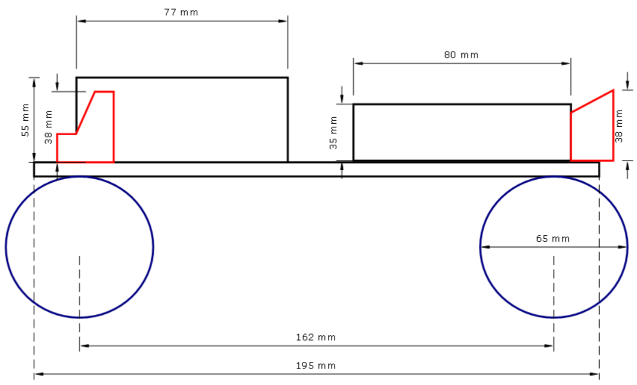

Fig. 3. Mechanical design front view

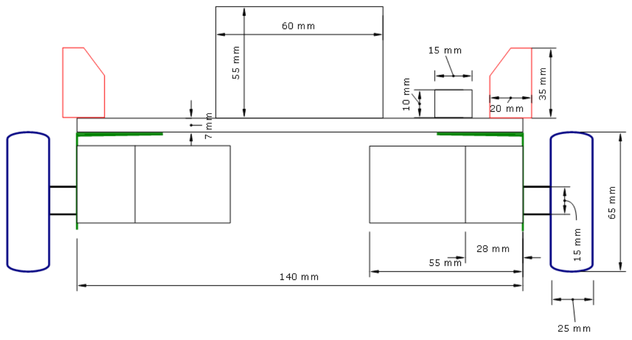

Fig. 4. Mechanical design back view

II. Assembly

In this chapter, the vehicle’s assembly and progress will be shown, step-by-step.

Some of the components before the installation:

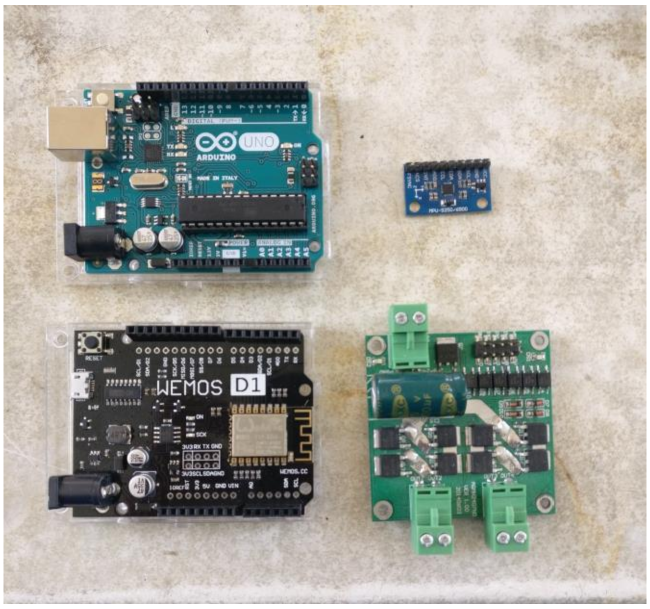

Fig. 5. Arduino Uno, MPU-9250, Wemos D1 R2, 7A Dual Motor Driver

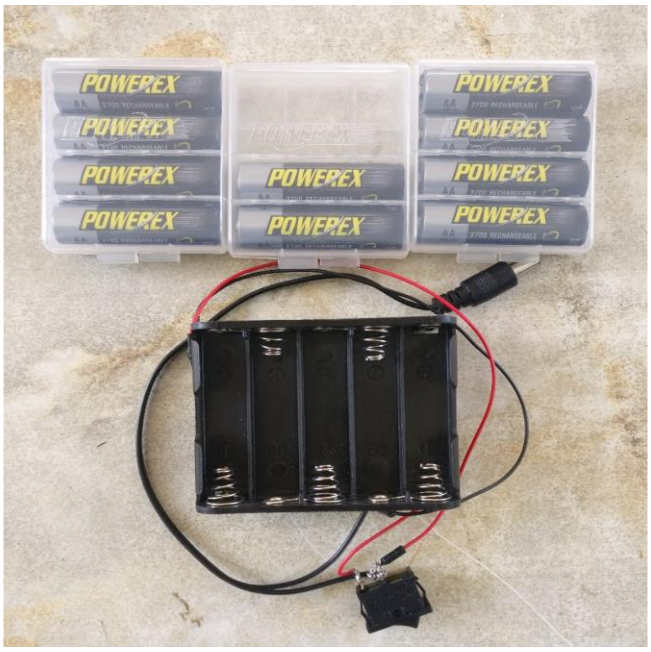

Fig. 6. Rechargeable batteries, battery pack, power supply wires

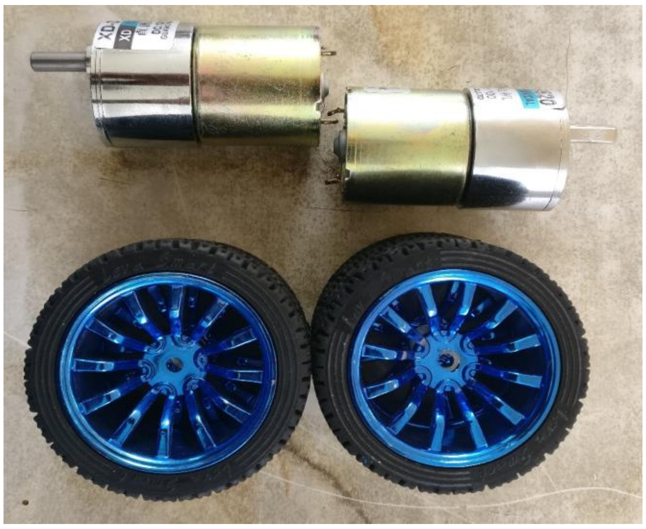

Fig. 7. DC Motors, Plastic wheels

First, I drilled some holes onto the base of the vehicle then screwed the motors' mounts and the axis mount as shown below:

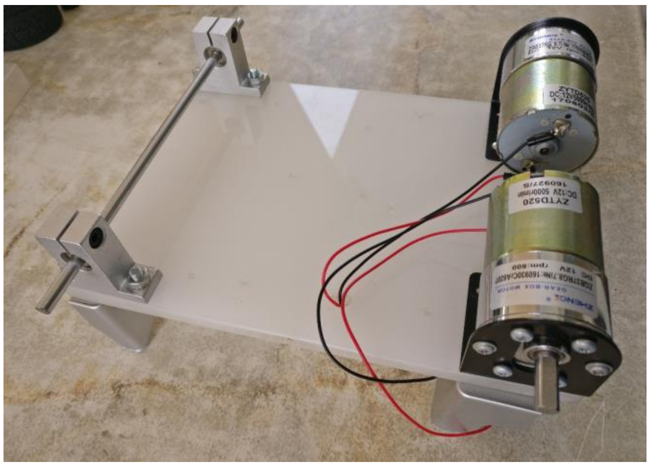

Fig. 8. Assembly 1

Then the wheels, foam layer and a couple of electronics were installed:

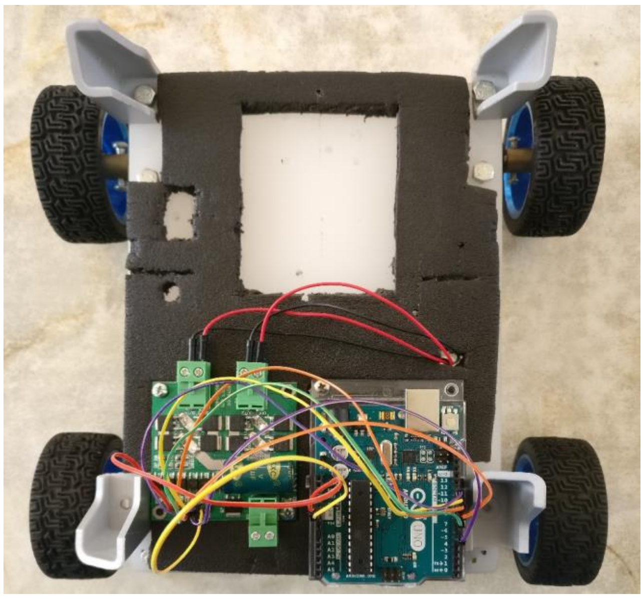

Fig. 9. Assembly 2

And last the final state of the car fully functioning at 1510 gr:

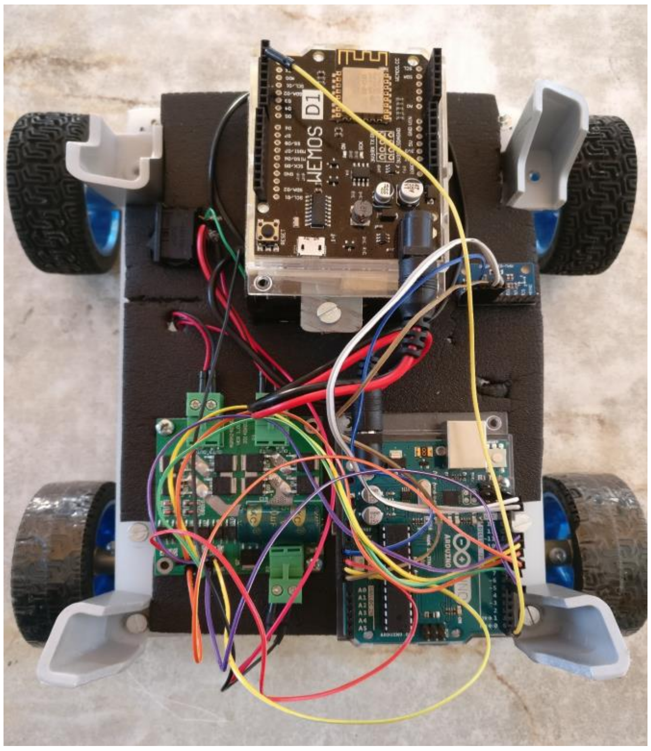

Fig. 10. Assembly final result

III. PROGRAMMING

The flow chart based on the source code is shown below:

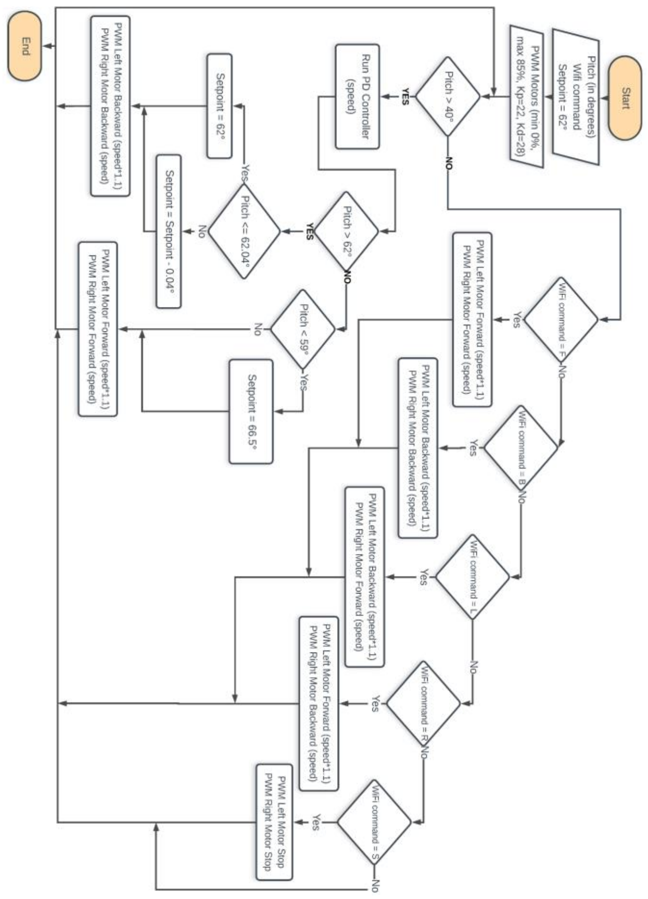
Start
Pitch (in degrees)
Wifi command
Setpoint = 62°
PWM Motors (min 0%, max 85%, Kp=22, Kd=28)
Pitch > 40°
YES
NO
Run PD Controller (speed)
Pitch > 62°
Pitch <= 62.04°
Setpoint = 62°
Setpoint = Setpoint - 0.04°
Pitch < 59°
Setpoint = 66.5°
PWM Left Motor Backward (speed*1.1)
PWM Right Motor Backward (speed)
PWM Left Motor Forward (speed*1.1)
PWM Right Motor Forward (speed)
WiFi command = F
WiFi command = B
WiFi command = L
WiFi command = R
WiFi command = S
PWM Left Motor Backward (speed*1.1)
PWM Right Motor Forward (speed)
PWM Left Motor Forward (speed*1.1)
PWM Right Motor Backward (speed)
PWM Left Motor Stop
PWM Right Motor Stop
Yes
No
End

Fig. 11. Flow chart

IV. Conclusion & Possible Improvements

The goal of this project has been achieved, but there is always space for improvement. The Wi-Fi is working flawlessly without latency, so I will mention ways to better the balancing system. The most helpful upgrade on the vehicle would be stronger and more precise DC motors, the ones used on this project run at 5000 rpm normally and have 10:1 gearbox so the shaft turns at 500 rpm when no load is present. The main problem is that they are a little loose when changing movement direction, this creates major disturbances on the tilt and on the PD controller behavior. A pair of DC motor with feedback would solve this problem. Also the positioning of the back wheel could contribute to the balancing. If the wheels were not under the base but in extension of the car's body, the center of balance would be at exactly 90° compared to the ground. Another idea would be to change the directional system. The addition of the latest directional technology using a little DC motor put on a geared axis, with each side having two joints would bring much smoother and natural turning experience.

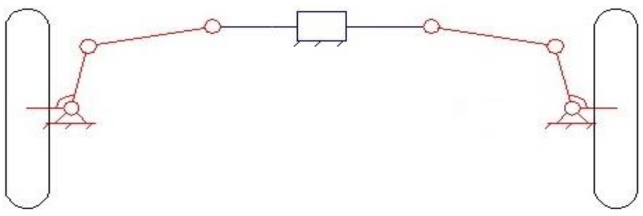

Fig. 12. Directional System

On a software level, I could use a second PD controller which would change the set point the first controller is trying to reach, so based on where the center of gravity is at the moment it would slightly change the first controller's set point. The center of gravity would be calculated by all the forces that are being applied on the vehicle minus the gravitational forces. Another approach would be having two different PD controllers for over the set point behavior and another one for under the set point behavior. In extension to the previous idea, two different PD controllers could be used for low and high feedback situations, the low feedback would be less sensitive for small movements means < ±3° and when the tilt is over ±3° a more sensitive and powerful feedback takes over.

V. Acknowledgements

Authors would like to acknowledge the University of West Attica postgraduate program of studies "MSc in Industrial Automation" for supporting this research project.

References

[1] Kris Winer, GitHub - kriswiner/MPU9250: Arduino sketches for MPU9250 9DoF with AHRS sensor fusion. Available at: https://github.com/kriswiner/MPU9250 [Accessed 15/02/2017]

[2] Ryan Downing, AutoPID | Arduino AutoPID library. Available at: http://ryandowning.net/AutoPID/ [Accessed 27/02/2017]

[3] Juraj Andrassy, GitHub - jandrassy/UnoWiFiDevEdSerial1: Implements Serial1 of Arduino Uno WiFi Developer Edition board to access the on-board ESP8266. Available at: https://github.com/jandrassy/UnoWiFiDevEdSerial1#uno-wifi-developer-edition-serial1 [Accessed 28/04/2017].

[4] Algorithm for self-balancing robot | Frobot – a DIY robot. Available at: https://frobotme.wordpress.com/2014/04/29/algorithm-for-self-balancing-robot/#comments [Accessed 30/04/2017]

[5] ESP8266WiFi library — ESP8266 Arduino Core 2.4.0 documentation. Available at: http://arduino-esp8266.readthedocs.io/en/2.4.1/esp8266wifi/readme.html [Accessed 30/04/2017]

[6] ESP8266: ESP8266WebServer Class Reference. Available at: https://links2004.github.io/Arduino/d3/d58/class_e_s_p8266_web_server.html [Accessed 01/05/2017]

[7] ForceTronics: How to Build an Android App to Control Your WiFi Enabled Arduino. Available at: http://forcetronic.blogspot.com/2016/08/how-to-build-android-app-to-control.html [Accessed 01/05/2017]