University Carlos III of Madrid

Industrial Electronics and Automation Engineering

Bachelor Thesis

Design, construction and programming of a low cost, Open Source robot for assistive activities

Author
Alvaro Ferrán Cifuentes

Supervisor
Juan González Víctores

Contents

List of Figures

Part I
Introduction

1 Introduction

2 Socio-economic factors

3 Regulatory compliance

4 Scope of the project

5 Project phases

[include Gantt diagram]

Part II
State of the art

Part III
Proposed Solution

6 Project components

6.1 3D Printer

3D printers are Computer Numerical Control (CNC) machines that are capable of transforming virtual 3D models created with a Computer Aided Design (CAD) software into real-world objects.

Created in 1984 by Chuck Hull of 3D Systems Corp this technology was little-known to the general public and was mainly used in industries for short runs of difficult pieces.
In 2005 Dr. Adrian Bowyer, from the University of Bath, UK, started the RepRap project. Its goal was "to produce a pure self-replicating device not for its own sake, but rather to put in the hands of individuals anywhere on the planet, for a minimal outlay of capital, a desktop manufacturing system that would enable the individual to manufacture many of the artifacts used in everyday life"

Today a vast range of 3D printers co-exist, varying in size, price and materials used.

→ [RepRap, Zcorp, chocolate, liquid sint, micro, house building]

→ [table with different methods?]

In this theses a RepRap Prusa Air 2 model is used. It is of a "fused filament fabrication additive manufacturing" type. This type of printers extrude mainly ABS or PLA plastics, and deposit new liquified material over ther previous layer, now solid, effectively building parts from the bottom up layer by layer.
→ **Terminal Imprusión Autorreplicante (T.I.A)** (add info + pic) Built within the scope of the Clone Wars project

6.2 Software

This type of 3D printers work by turning 3D models into plastic parts. These models are first modelled in a CAD program and then processed with a *slicing* software to divide the model into layers of G-code, which is the standard language interpreted by CNCs. This is then introduced in a third piece of software which feeds it to the printer.

6.2.1 3D Modelling

In this project Sketchup has been used to create the printed parts. Owned by the company Trimble Navigation it is a WYSIWYG (What You See Is What You Get) modelling editor with a large online warehouse of parts available for download.

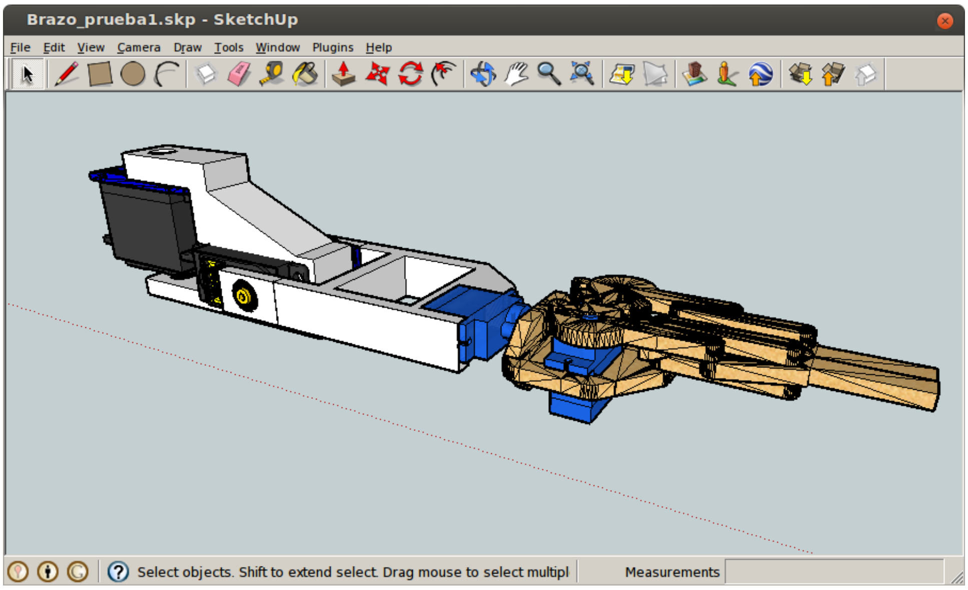

Figure 1: SketchUp software

In order to make it compatible with the slicing sofware, Sketchup's propietary format, *SKP*, has to be converted to the standard *STL*. In order to accomplish this the *Su2stl.rb* plugin is installed. A new *Plugins* menu appears in Sketchup which contains the Import/Export options, where the desired output format and model units are specified.

6.2.2 G-Code Generator

Once the model is converted to *STL* it then has to be sliced. Since 3D printers work by building layer upon layer of plastic, the model has to be transformed into the same format. The G-code generator converts the CAD model into layers of CNC instructions. There are three main slicing programs, each with their own benefits:

- Skeinforge:
  The first slicing program used in homemade 3D printers. It is by far the most complete of the three. It allows the user to control each and every imaginable setting of the printer, from the axis' speeds to the retraction distance of the plastic into the extruder while moving. However, because of this it has a very steep learning curve which makes it unsuitable for the average consumer.

- Slic3r:
  Slic3r was created as an user-friendly software, which only gives the final user a choice in the basic settings, such as printing speeds, filament widths or part infills. As a result it is an easier program to slice parts with a sufficient level of customization. It has nonetheless problems converting models with imperfections or broken shapes.

- Cura
  Finally, Cura is also designed with user-friendliness in mind. This slicer is more robust than Slic3r, in that it will accept models with imperfections, and will try to correct them. It also features a box simulating the print area in which the model can be moved around, turned or scaled before printing. This last feature is specially useful if minor changes need to be made, without returning to the CAD software.

6.2.3 CNC Controller

6.3 Li-Ion Battery

The whole system is powered by a lithium ion 12V 6800mAh battery.

Figure 2: Li-ion 12V 6800mAh battery with charger

Many different types of batteries are available in the market, each with their own benefits and disadvantages. The most common types are alkaline, lithium ion (Li-ion), lithium polymer (LiPo), lead acid, nickel–metal hydride (NiMH) and nickel–cadmium (NiCd).

	NiCd	NiMH	Lead Acid	Li-ion	Li-ion polymer	Reusable Alkaline
Gravimetric Energy Density(Wh/kg)	45-80	60-120	30-50	110-160	100-130	80 (initial)
Internal Resistance (includes peripheral circuits) in mΩ	100 to 200[1] 6V pack	200 to 300[1] 6V pack	<100[1] 12V pack	150 to 250[1] 7.2V pack	200 to 300[1] 7.2V pack	200 to 2000[1] 6V pack
Cycle Life (to 80% of initial capacity)	1500[2]	300 to 500[2,3]	200 to 300[2]	500 to 1000[3]	300 to 500	50[3] (to 50%)
Fast Charge Time	1h typical	2-4h	8-16h	2-4h	2-4h	2-3h
Overcharge Tolerance	moderate	low	high	very low	low	moderate
Self-discharge / Month (room temperature)	20%[4]	30%[4]	5%	10%[5]	~10%[5]	0.3%
Cell Voltage(nominal)	1.25V[6]	1.25V[6]	2V	3.6V	3.6V	1.5V
Load Current - peak - best result	 20C 1C	 5C 0.5C or lower	 5C[7] 0.2C	 >2C 1C or lower	 >2C 1C or lower	 0.5C 0.2C or lower
Operating Temperature(discharge only)	-40 to 60°C	-20 to 60°C	-20 to 60°C	-20 to 60°C	0 to 60°C	0 to 65°C
Maintenance Requirement	30 to 60 days	60 to 90 days	3 to 6 months[9]	not req.	not req.	not req.
Typical Battery Cost (US$, reference only)	$50 (7.2V)	$60 (7.2V)	$25 (6V)	$100 (7.2V)	$100 (7.2V)	$5 (9V)
Cost per Cycle(US$)[11]	$0.04	$0.12	$0.10	$0.14	$0.29	$0.10-0.50
Commercial use since	1950	1990	1970	1991	1999	1992

Figure 3: Table comparing different battery technologies

6.4 Voltage level converters

6.4.1 DC-DC Step-Down Converter

Voltage level converters are used to adapt a source's voltage to that required by the load. In this thesis a DC-DC converter is used to decrease the 12V given by the battery to the 5V required by the logic components as well as the servomotors.

The simplest method would be to use a linear regulator such as a 7805, which is a cheap, single-component solution. However, this is greatly inefficient solution, since a great part of the power is dissipated as heat. For instance, if a 7805 were to be used in this project, about $\frac{Power_{in} - Power_{out}}{Power_{in}} = \frac{12 \cdot I - 5 \cdot I}{12 \cdot I} = \frac{12-5}{12} = 58.33\%$ of the power is wasted.

A much more efficient solution is to use a switching regulator such as a Buck converter, which has an efficiency level of around 95% [1]. Buck converters work by switching rapidly between "On" and "Off" states, which sets the output voltage in function of the duty cycle $d = \frac{time_{on}}{time_{on} + time_{off}}$. The converter used includes a potentiometer to set the output level by selecting the duty cycle.

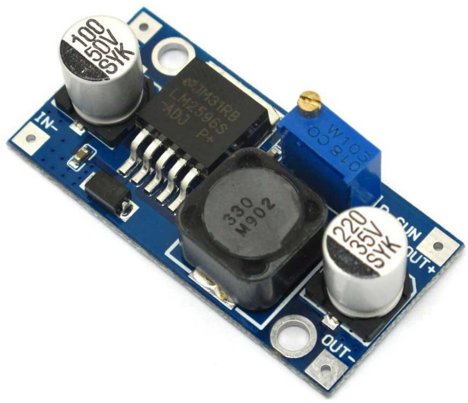

Figure 4: LM2596S step-down converter

The converter's electrical specifications are:

- Adjustable input voltage: 3.2 - 40V
- Adjustable output voltage: 1.25 - 35V ($V_{in} > V_{out}$ + 1.5V)
- Max. output current: 3A

6.4.2 Bidirectional logic level converter

In order to enable serial communication between the Arduino and the Raspberry Pi another voltage level converter must be introduced, since the former operates at a 5V level while the latter does so at a 3.3V level.

In this case a switching regulator like the previous one will not work because the communications are much faster than the regulator's switching speed. Therefore a bidirectinal, low power converter can be built out of transistors.

Figure 5 shows a simple converter model. Analyzing the circuit from the low side:

- If a logic one is emitted, the transistor source pin is grounded and it switches on, pulling down the high side to zero.
- If a logic zero is sent, the transistor is tied high and so is off, leaving the high pin connected to the pull-up resistor and thus seeing a one.

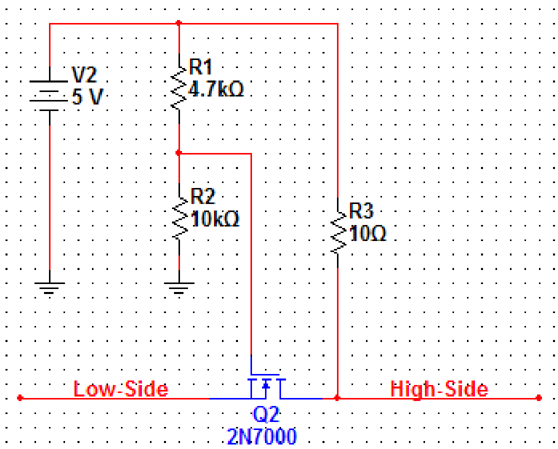

Figure 5: Bidirectional voltage converter

This setup works for one line, two identical circuits are needed in order to provide for serial communication. For this project a commercial board is used to reduce the total size of the converter by using SMD components.

Figure 6: JY-MCU 5V-3V converter

This board is used with UART communication, but is equally adequate for I^{2}C, SPI or one-wire communication.

6.5 Motors

6.5.1 DC motor

Figure 7: GA25Y370 motor

6.5.2 Servomotors

Figure 8: GOTECK GS-551MG servo

Figure 9: TowerPro SG90 servo

6.6 Arduino

Arduino is a family of low-cost electronic boards designed to be easily programmable. From the official Arduino website, "Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. "

Arduino is programmed using its own language, which is merely a set of C/C++ functions compiled with *avr-g++*. They can nonetheless be programmed in pure C or C++ in an external IDE and have code uploaded to it as any other AVR board.

In this proyect an Arduino Nano v3 with an ATmega 328 microcontroller has been chosen mainly due to its processing power and reduced size.

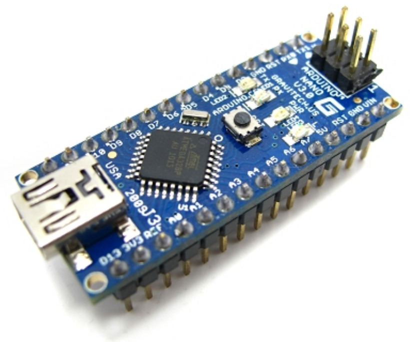

Figure 10: Arduino Nano v3

The official Arduino Nano V3 specifications are:

- **Microcontroller:** Atmel ATmega168 or ATmega 328
- **Operating Voltage (logic level):** 5V
- **Input Voltage (recommended):** 7-12V
- **Input Voltage (limits):** 6-20V
- **Digital I/O Pins:** 14 (of which 6 provide PWM output)
- **Analog Input Pins:** 8
- **DC Current per I/O Pin:** 40mA
- **Flash Memory:** 16 KB (ATmega168) or 32 KB (ATmega328), of which 2 KB used by bootloader
- **SRAM:** 1 KB (ATmega168) or 2 KB (ATmega328)
- **EEPROM:** 512 bytes (ATmega168) or 1 KB (ATmega328)
- **Clock Speed:** 16MHz
- **Dimensions:** 0.73" x 1.70"
- **Communications:** UART, SPI and I^2C buses

6.7 Raspberry Pi

From the official website of the homonymous foundation, the Raspberry Pi is a "credit-card sized computer that plugs into your TV and a keyboard. It is a capable little computer which can be used in electronics projects."

Figure 11: Raspberry Pi model B

Available in two models, A and B, the Raspberry has a Broadcom BCM2835 System On a Chip, which includes an ARM1176JZF-S 700MHz processor and a VideoCore IV GPU. It includes as well a 256Mb RAM, upgraded to 512Mb in model B.

The Pi features:

- HDMI, composite and raw DSI video outputs
- 3.5mm audio jack
- SD card socket
- Low-level peripheral connections including:
  - 8 General Purpose Input Output (GPIO) pins
  - Universal Asynchronous Receiver Transmitter (UART) bus
  - Inter-Integrated Circuit (I^2C) bus
  - 2 Serial Peripheral Interface (SPI) buses
  - Power pins: 3.3V, 5V and GND
- Ethernet socket
- USB hub (1 socket in model A, 2 in model B)

The main storage unit is the SD card, and that is where the OS is flashed, normally a Linux distribution. The most popular is Raspbian, an adapted version of Debian Wheezy, although other Linux distros or even other OS like Android or XBMC can be used.

(a) WiFi USB dongle (b) PlayStation 2 EyeToy (c) LCD screen 16x2

Figure 12: Raspberry Pi peripherals

In this proyect a Raspberry Pi model B running Raspbian manages the software side of the robot. It has an EDUP 802.11n WiFi USB dongle , a PlayStation 2 EyeToy USB camera and a 16x2 character LCD screen connected in order to create a WIFI Access Point, stream video to the user and signal its status respectively.

6.8 Android Phone

Android is... blablabla
for this proyect the model bla bla haas been used

Figure 13: Android smartphone Haipai Noble H868

7 Assembly

7.1 Mechanical structure

7.2 Electrical connections

8 Arduino

8.1 Objective

The Arduino is designed to control a relatively high number of devices. While it is a standard code has the following funct() bla bla...

8.2 Code

In this section the code written for the robot's controller will be explained. A flowchart diagram of the program is illustrated by Figure 14.

As it can be seen, the Arduino first defines all the robot's data, including the motor, servomotor and communication pins. This ensures the microcontroller knows where to send each datum once the user connects to the robot.

The program then enters its Loop function. Here it will check if the serial port is available, eg the user has sent a stream of data. Once the port is available, the Read function is called, which stores every byte received into a string to be used later. Once the reading has ended the code checks if it has received a special end-of-line character that signals the end of the data stream. If all the data was retrieved the code moves on to the next function.

The Parse function is called upon next. This function's purpose is to break and convert the previously stored string into the corresponding variables needed by each element, taking into account their sizes and types. Hence, it transforms one line of numbers into many parameters such as rotation angle, arm selection or movement direction which will be used by the next function.

With the data correctly formatted, the program executes the Process function which is where the "thinking" is done. Here are defined all the rules the robot must follow, such as knowing which claw to close depending on the side chosen by the user but closing both if the symmetry box was checked. It takes the data provided by the previous function and processes them to end up with a structured list of variables ready to be assigned to each element.

In the next step the Write function is called. This very simple function goes through the previous list assigning each variable to the corresponding element's assigned pins.

Finally, the code clears the initial string to make space for new data, resets the flag informing of the correct retrieval from the serial port and proceeds to the next iteration within the Loop function, restarting the process.

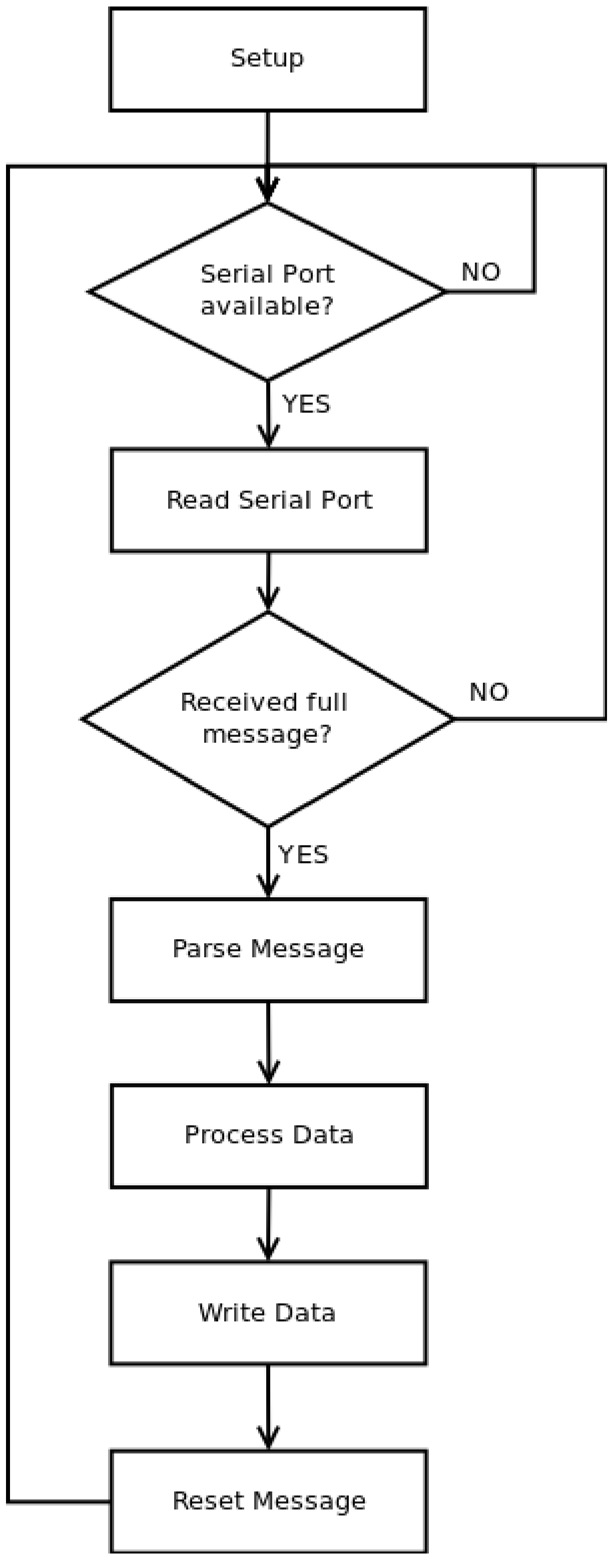

Figure 14: Arduino program flowchart

9 Raspberry Pi

9.0.1 WiFi Acces Point

9.0.2 MJPG Streamer

9.0.3 IP/UART Bridge

10 Android

Part IV
Conclusion

References

[1] EXAMPLE CITATION: Leslie Lamport, *LaTeX: a document preparation system*. Addison Wesley, Massachusetts, 2nd edition, 1994.