



M 2014

MOTORIZED MICROSCOPE STAGE FOR SMARTPHONE

JOÃO CARLOS VISEU OLIVEIRA

DISSERTAÇÃO DE MESTRADO APRESENTADA

À FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO EM

MESTRADO INTEGRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

A Dissertação intitulada

“Motorized Microscope Stage for Smartphone”

foi aprovada em provas realizadas em 18-07-2014

o júri


Presidente Professor Doutor Paulo José Cerqueira Gomes da Costa
Professor Auxiliar do Departamento de Engenharia Eletrotécnica e de Computadores
da Faculdade de Engenharia da Universidade do Porto


Professor Doutor André Vaz da Silva Fidalgo
Professor Adjunto do Departamento de Engenharia Eletrotécnica da Instituto
Superior de Engenharia do Porto


Professor Doutor Dirk Christian Elias
Professor Catedrático Convidado do Departamento de Engenharia Eletrotécnica e de
Computadores da Faculdade de Engenharia da Universidade do Porto

O autor declara que a presente dissertação (ou relatório de projeto) é da sua exclusiva autoria e foi escrita sem qualquer apoio externo não explicitamente autorizado. Os resultados, ideias, parágrafos, ou outros extratos tomados de ou inspirados em trabalhos de outros autores, e demais referências bibliográficas usadas, são corretamente citados.


Autor - João Carlos Viseu Oliveira

Faculdade de Engenharia da Universidade do Porto



Motorized Microscope Stage for Smartphone

João Carlos Viseu Oliveira

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Dirk Christian Elias

Co-Supervisor: Carlos João Rodrigues Costa Ramos

Co-Supervisor: Jorge Manuel de Matos Reis

July 28, 2014

Abstract

Every year malaria kills millions of people because the time between a blood test and the result is too much to allow an effective treatment. To respond to this limitation Fraunhofer AICOS presented the idea of creating an fast, cheap and autonomous system, fully replicable in third world countries, and capable of giving a preliminary result to this test by analysing 100 spots in the blood smear (as recommended by World Health Organization). This translated to a project where a smartphone was used as the brain and controlled a XY Table while taking the necessary pictures of the smear. This thesis focused in the table part. It needed to be capable of under $500\mu m$ steps in each direction and be fully powered by the USB connection of the smartphone.

Several actuators were studied and tested and two solutions were presented: one based in piezoelectric benders and other based in common DC motors. Both solutions feature Arduino based electronic drivers and are fully controlled by an Android smartphone application, components fully developed during this thesis. To allow the replication, a 3D printer was used to produce most of the parts for the solution.

The DC motor solution used old CD drives and was capable of steps under $300\mu m$. The piezoelectric solution featured a novel design fully 3D printed and featured steps below $250\mu m$.

An ingenious method to measure the steps was used. It was composed by an high DPI optical mouse and a computer software that measured the displacement as pixels on the screen.

This research project was fully successful and proved that the idea is possible and valid. It presented two different valid paths with its pros and cons complying with all objectives and requirements bringing a lot of value to the host company and to the whole project.

Agradecimentos

Antes de mais nada quero agradecer à Fraunhofer AICOS e um dos seus diretor e meu orientador, o professor Dirk Elías por me terem dado esta oportunidade de me integrar numa empresa de referência na área de investigação e me terem dado uma inestimável experiência que certamente pesará bastante no meu futuro.

Quero agradecer também a toda a equipa da Fraunhofer AICOS que me tratou como se fosse da casa e principalmente aos cientistas Luís Rosado, que sempre disponibilizou o seu tempo quando necessário, e Manuel Monteiro que me foi aturando ao longo de todo processo, ajudando-me em tudo o que foi necessário.

Agradeço também ao professor Carlos Ramos, meu co-orientador, que me disponibilizou o seu tempo e inegável experiência no mundo da eletrónica apontado-me o caminho a seguir.

Deixo também um sincero agradecimento ao Engenheiro Jorge Reis pela paciência e sincera boa vontade que demonstrou ao aceitar ajudar-me apesar da sua vida bastante ocupada, pela sua infinita fonte de conhecimento e pela valiosa ajuda prestada que, sem ela, o projeto não chegaria onde chegou. Fica apenas o amargo na boca de não o ter contactado mais cedo, certamente o resultado seria ainda melhor.

Agradeço também à empresa Igus que me disponibilizou o material necessário para testar algumas das ideias apresentadas.

Agradeço sinceramente e o mais humildemente que consigo aos meus pais que sempre me deram tudo e, principalmente, me deram esta possibilidade de me instruir e trabalhar a fazer o que mais gosto. É uma grande felicidade puder deixá-los orgulhosos e felizes pelo meu sucesso. Não posso também esquecer a minha irmã, um das raparigas mais fortes e trabalhadoras que conheço. Sem ela, quem me distrairia durante a escrita desta dissertação? Tenho de agradecer também à minha namorada que aturou todos os meus humores e devaneios sem pestanejar e sempre me deu todo o apoio que precisei e ainda mais que isso. Sem ela, certamente, não estaria onde estou.

Como prometido, deixo também o meu agradecimento à minha tia que me aturou e leu tudo de fio a pavio à procura das inúmeras gralhas inseridas pelas noites de escrita.

Deixo também uma palavra de apreço a todos os verdadeiros amigos que me acompanharam ao longo deste percurso, alguns adquiridos ao longo do caminho, outros amigos à tanto tempo que nem me lembro de lá não estarem, e me foram dando a força necessária para continuar, no meio de jantaras e cafezadas no DEEC.

Não poderia terminar e esquecer a instituição que me instruiu e desenvolveu ao longo destes 5 anos. Deixo então os meus sinceros agradecimentos à FEUP e a todos os professores que me acompanharam desde o primeiro ano em que não passava de um miúdo com sonhos até agora onde continuo um miúdo com sonhos mas um bocado mais conhecedor do mundo que me rodeia.

João Oliveira

“Does this unit have a soul?”

Mass Effect 3

Contents

Agradecimientos	v
1 Introduction	1
1.1 Motivation and Context	1
1.2 Project Presentation	2
1.2.1 Requirements, Guidelines and Limitations	3
1.3 Objectives Proposed	5
1.4 Thesis structure	5
2 State of Art	7
2.1 XY-Stages	7
2.1.1 What are XY-Stages?	7
2.1.2 Implementing XY-Stages	8
2.1.3 Remarks	19
2.2 Piezoelectric Technology	20
2.2.1 Historic Review	21
2.2.2 Piezoelectric Effect	21
2.2.3 Basic Piezoelectric Actuators	23
2.2.4 Piezoelectric Actuators and Motors	25
2.2.5 How to drive a Piezoelectric Actuator?	29
2.2.6 Remarks	30
2.3 Stepper Motor Technology	31
2.3.1 Historic Review	31
2.3.2 Types and Operation	31
2.3.3 How to drive a Stepper Motor?	33
2.3.4 Remarks	35
2.4 Piezoelectric Benders vs Stepper Motors	36
2.5 Mechanical systems	37
2.5.1 CD/DVD computer drive	37
2.5.2 Printers	38
2.5.3 Drylin Linear system	38
2.6 Smartphone - Arduino Communication	39
3 Electronic Driver	41
3.1 Piezoelectric Bender Driver	41
3.1.1 Theoretical Study	41
3.1.2 Implementation	51
3.1.3 Experimental Results	60

3.1.4	Remarks	64
3.2	DC Motor Driver	66
3.2.1	Theoretical Study	66
3.2.2	Implementation	67
3.2.3	Experimental Results	69
3.2.4	Remarks	72
3.3	Conclusion	73
4	XY Table	75
4.1	Principle Theory	75
4.1.1	Implementation premisses	76
4.2	3D printer	77
4.2.1	Materials	79
4.2.2	Calibration and Tests	79
4.3	Triangular Tip topology	81
4.3.1	Model Development	81
4.3.2	Physical Implementation	83
4.3.3	Experimental Results	85
4.3.4	Remarks	86
4.4	Claw/Lever Tip topology	87
4.4.1	Model Development	87
4.4.2	Physical Implementation	89
4.4.3	Experimental Results	90
4.4.4	Remarks	90
4.5	DC motor table topology	91
4.5.1	Physical Implementation	93
4.5.2	Experimental Results	94
4.5.3	Remarks	98
4.6	Three bender table topology	98
4.6.1	Model Development	98
4.6.2	Physical Implementation	103
4.6.3	Experimental Results	105
4.6.4	Remarks	107
4.7	Conclusion	107
5	Smartphone Controller Library	109
5.1	Technologies Used	109
5.1.1	Android Technologies	109
5.1.2	Arduíno Technologies	111
5.2	Mandatory Functions	112
5.3	Communications Protocol	113
5.4	Android Library	114
5.4.1	USB Serial for Android	114
5.4.2	MicroStage Serial Library	118
5.5	Arduíno Firmware	125
5.5.1	setup()	125
5.5.2	loop()	126
5.6	Test application	128
5.7	Conclusion	132

6	Conclusion and Future Work	135
6.1	Conclusion	135
6.2	Future Work	137
6.2.1	Electronic Driver	137
6.2.2	XY Table	138
6.2.3	Smartphone Controller Library and Application	138
	References	141

List of Figures

1.1	Fraunhofer's conceptual idea.	2
1.2	Block diagram of the concept.	3
2.1	M-545 XY-Stage from Physik Instrumente.	8
2.2	PZT Stack actuator [1].	9
2.3	PZT Stack driven stage topology [1].	9
2.4	PZT Stack driven stage hinges [1].	10
2.5	Open loop (a) and closed-loop (b) response to step motion [1].	10
2.6	PZT Worm actuator step sequence [2].	11
2.7	PZT Worm Motor concept [2].	12
2.8	PZT Worm driven stage example results [2].	12
2.9	Electrostatic Comb-drive actuator [3].	14
2.10	Electrostatic Comb actuator driven stage topology [3].	15
2.11	Electrostatic Comb actuator driven stage results [3].	15
2.12	Stepper Motor [4].	16
2.13	Stepper Motor driven stage topology [5].	17
2.14	Stepper Motor driven stage results [5].	18
2.15	Piezoelectric interactions [6].	21
2.16	Direct piezoelectric effect. a) Normal molecule. b) Formation of a dipole. c) Effect of an external force in the material [6].	22
2.17	PZT polarization curve [6].	22
2.18	Force versus Deflection of PZT actuators. a) Individual actuator analysis. b) Different behaviours of the 3 possible assemblies [7].	23
2.19	PZT actuator equivalent circuit [7].	23
2.20	PZT stack actuator by Pi Ceramic.	24
2.21	PZT bender actuator by Pi Ceramic.	25
2.22	Simple PZT stack displacement amplifier.	26
2.23	Diamond actuator system by Noliac [8].	26
2.24	Hydraulic amplification system.	26
2.25	Stick and slip principle by PI [7].	27
2.26	Friction slide principle.	27
2.27	Worm motor actuation cycle.	28
2.28	Ultrasonic motor example.	28
2.29	Power Driver for a serial PZT bender.	29
2.30	Step-up converter.	30
2.31	Power Driver for a parallel PZT bender.	30
2.32	Stepper motors classes.	31
2.33	Variable Reluctance stepper motor.	32

2.34	Claw Poles stepper motor.	32
2.35	Hybrid stepper motor.	33
2.36	Unipolar stepper motor.	33
2.37	Bipolar stepper motor.	34
2.38	Stepper motor driver block diagram.	34
2.39	Unipolar stepper power driver circuit.	34
2.40	Bipolar stepper power driver circuit.	35
2.41	CD drive examples.	37
2.42	Printer example. Picture from HP Inkjet mechanism patent.	38
2.43	Miniature Drylin T systems. Picture by Igus.	39
2.44	Example application results.	40
3.1	BA4510 PZT bender.	42
3.2	Proposed driver circuit diagram.	43
3.3	1 - Coil charging, 2 - Capacitor charging.	44
3.4	Coil current waveform	44
3.5	Capacitor current waveform	46
3.6	PSIM converter model.	48
3.7	PWM waves. V_{cont_1} is the PWM for the first stage and V_{cont_2} is the PWM for the second one.	49
3.8	Coil currents. I_{l_1} is the current from the first coil and I_{l_2} is the current from the second one.	49
3.9	Capacitor currents. I_{c_1} is the current from the first capacitor and I_{c_2} is the current from the second one.	49
3.10	Output waveforms. V_{o_1} is the output voltage of the first stage, V_{o_2} is the output voltage of the second one and I_{out} is the output current.	50
3.11	A simple voltage divider.	52
3.12	Semiconductor preferences according International Rectifier [9].	53
3.13	Mixed channels MOSFET h-bridge.	54
3.14	Driver schematics.	57
3.15	PCB layers.	58
3.16	Stripper Boost Converter PCB.	59
3.17	Boost Converter.	60
3.18	Wave response of the driver.	62
3.19	Output voltage of the driver.	63
3.20	GSAM Battery Monitor.	64
3.21	Li-Ion battery discharge plot [10].	65
3.22	PZT bender driver consumption plots.	65
3.23	DC motor h-bridge.	66
3.24	L293E h-bridge circuit.	67
3.25	DC motor driver circuit.	69
3.26	DC motor driver.	70
3.27	L293E step waveform. Notice the 5V peak voltage.	71
3.28	DC motor driver consumption plots.	72
4.1	Same figure presented in 2.2.4.2, see 2.25	76
4.2	Studied rubbers.	78
4.3	Makerbot Replicator 2x.	78
4.4	Build plate spring screws.	80

4.5	Failed cube print.	80
4.6	New glass base.	81
4.7	Cube comparison. Notice the curled bases and the improved resolution of the red cube.	81
4.8	Google Sketchup 3D model of the triangular tip topology motor.	82
4.9	Lever theory applied to the triangular tip topology.	82
4.10	Google Sketchup 3D model of the triangular tip topology.	83
4.11	Triangular Tip topology printed pieces.	84
4.12	Sample printed piece.	85
4.13	Triangular tip topology assembled system.	85
4.14	Applied voltage to the actuator. Note the dead zone of the converter.	86
4.15	Igus drylin T miniature system.	87
4.16	Google Sketchup 3D model of the claw/lever tip topology actuation system.	88
4.17	Google Sketchup 3D model of the claw/lever topology using an Igus system model.	89
4.18	Claw/lever topology assembled system.	89
4.19	Applied voltage to the actuator.	90
4.20	Example cd-rom system.	91
4.21	Grid movement implementation for the DC motor topology. The values represent a cell position.	92
4.22	DC motor XY Table with a microscope slide.	93
4.23	Google Sketchup model of Y Axis support.	94
4.24	Mousotron screenshot. The "X Coord" and "Y Coord" are important fields.	95
4.25	System assembled to measure the DC motor topology displacement. A black card-board was used to enable the measure.	96
4.26	DC motor topology validity test.	96
4.27	Position of the DC motor table.	97
4.28	DC motor topology resolution test step sizes.	97
4.29	Third bender circuit.	99
4.30	Three bender table, top view without the moving platform.	100
4.31	Three bender table, side view.	100
4.32	Three bender table y axis principle.	101
4.33	Three bender table x axis principle.	102
4.34	Google Sketchup three bender table.	102
4.35	Grid movement implementation for the Three bender topology. The values represent a cell position and can be compared with figure 4.21.	103
4.36	Assembled three bender table.	103
4.37	Detail of one of the tip.	104
4.38	Detail of the small additions to the design.	104
4.39	System assembled to measure the three bender topology displacement.	106
4.40	Position of the three bender table.	106
4.41	Three bender step sizes.	107
5.1	Eclipse with ADT plugin.	110
5.2	Samsung Galaxy S4 Zoom. Foto from GSMarena.com.	110
5.3	Arduino IDE.	111
5.4	Arduino Leonardo. Foto from Arduino.cc.	112
5.5	Master packets model.	114
5.6	A simple request chain. This is just an example, the OPEN and CLOSE transactions just need to be made once, not at every transaction.	116

5.7	High level UML class diagram of the <i>USB Serial for Android</i> library.	117
5.8	High level UML class diagram of the <i>MicroStage Serial</i> library.	119
5.9	Structure of the DC motor topology loop().	126
5.10	DC motor topology FSM.	128
5.11	Structure of the PZT motor topology loop().	129
5.12	PZT motor topology FSM.	130
5.13	App welcome screen.	131
5.14	PZT motor controller screen.	131
5.15	DC motor controller screen.	132
5.16	XY table controller screen.	132
5.17	PZT motor control JAVA application running on a Windows PC.	133

List of Tables

2.1	PZT Stack driven stage pros and cons.	11
2.2	PZT Worm driven stage pros and cons.	13
2.3	Comb Actuator driven XY stage pros and cons.	16
2.4	Stepper Motor driven XY stage pros and cons.	18
2.5	Summary of all topologies shown.	20
2.6	Piezoelectric solution versus Stepper motor solution.	36
3.1	Piezoelectric driver ratings.	48
3.2	Piezoelectric driver capacitor and inductance values.	48
3.3	Component ratings.	53
3.4	Ordered components in Mouser Electronics.	55
3.5	H-bridge logic. Taken from the datasheet, property of Texas Instrument.	60
3.6	Components need for the DC motor driver if ordered in Mouser.	68
4.1	Measurements comparison.	84
5.1	Every request detailed.	115
5.2	Slave possible responses.	116
6.1	Proposed solutions pros and cons.	137

Abbreviations and Symbols

DC	Discrete Current
AC	Alternating Current
PZT	Piezoelectric
Javadoc	Java documentation
UML	Unified Modelling Language
DSP	Digital Signal Processor
PI	Proportional-Integral
USB	Universal Serial Bus
PCB	Printed Circuit Board
3D	Three Dimensions
PWM	Pulse-width modulation
I/O	Input/Output
OTG	On-the-Go
PSIM	PowerSim software
BJT	Bipolar Junction Transistor
MOSFET	Metal–Oxide–Semiconductor Field-Effect Transistor
IGBT	Insulated-Gate Bipolar Transistor
IC	Integrated Circuit
ADC	Analog-to-Digital Converter
ABS	Acrylonitrile Butadiene Styrene
PLA	Poly lactide
STL	Stereolithography
bps	bits per second
DPI	Dots Per Inch
DPmm	Dots Per millimetre
ADT	Android Development Tools
GUI	Graphic User Interface
XML	Extensible Markup Language
FSM	Finite-State Machine

Units of Measure

Hz	Hertz
A	Ampere
V	Volt
W	Watt
s	Second
g	Grams
F	Farad
H	Henry

Chapter 1

Introduction

1.1 Motivation and Context

According to World Health Organization, malaria is the second [11] leading cause of death in African continent. One of the main reasons for that is the time that it takes to get a result from a blood sample analysis (in the order of days). That time severely limits what any doctor can do in order to try to cure or save the patient, often leading to fatal results due to the late administration of an appropriate drug. Despite that great disadvantage today, the lack of viable alternatives (cost-wise) makes it the current best option to deal with malaria diagnosis.

One of the main fields for decreasing the percentage of deaths with malaria is the diagnosis:

“Despite an obvious need for improvement, malaria diagnosis is the most neglected area of malaria research, accounting for less than 0.25% (\$700,000) of the \$323million investment in research and development in 2004.” [12, Introduction]

If some method of cheap and fast diagnosis (at least, preliminary) was developed, it could really help saving millions of lives and change this paradigm. The main problem is the junction of the qualities cheap and fast.

Fast malaria diagnosis tools already exist as we can see in [12], named *Rapid Diagnose Test (RDT)* kits but they are cost prohibitive, averaging \$0.55 to \$1.50 per test. Multiplying this for millions of tests, we easily reach the conclusion that something else should be used. We must take account of the possibility of fake positives that represent 15% of all RDT tests [12] and the over-diagnosis can quickly decimate any pharmacy budgets.

Nowadays, the used diagnosis option is the microscope analysis. The estimated cost for each microscope test is \$0.12~\$0.40 [12] which represents an average of five times less the cost of RDT kits. But the main problem is that there are almost no laboratories near high-transmission areas leading to the problem enumerated before (the delay between the blood collection and obtaining the result).

Hereupon, the main idea is to eliminate that delay between the collection and the result using a portable system that can perform the analysis in the field and give a preliminary result. That idea is presented next.

1.2 Project Presentation

Fraunhofer Portugal is a non-profit private organization that aims to create scientific knowledge oriented towards the social well-being of its end-users. With that in mind, Fraunhofer was proposed a project inserted in ICT4D MalariaScope, a system that can perform an automated blood analysis. A part of this project (the electronic and mechanical) was inserted as a thesis proposal at FEUP.

Since this was a very specific problem, the project concept and its main objectives were easy to identify. We needed to develop a tool that could lead to a fast, preliminary and fully autonomous analysis of blood smears in order to give an approximate answer to the question: "Is the patient's blood infected with malaria?" in an understandable way for a person with basic training. It should be as cheap as possible and replicable without engineering knowledge.

With everything said above in mind, we sketched the following concept: to develop a motorized XY microscope stage driven by a smartphone and that will use its imaging processing abilities to perform the analysis.



Figure 1.1: Fraunhofer's conceptual idea.

The first thought that comes to mind is how will this perform to make a reliable analysis of the blood smear. The concept is really simple and can be explained by imagining a ten by ten grid on top of the blood smear. Each cell represents an image that the system will analyse. The total number of tests is one hundred and that is the minimum needed to provide a reliable answer according to World Health Organization [13]. The placement within the cell doesn't have to be really precise but it has to ensure that the analysis will be performed in one hundred different points. As a direct consequence, repeatability is optional too but if it can be implemented without any loss for the main objectives it would be a big plus.

For the physical movement, the intention is to use a small XY-Stage system based in piezo-electric or stepper motors with an electronic driver controlled by an Arduino based board. As said

above, the system will be controlled by a smartphone so this implies that a library will have to be developed in order to make possible the communication between the Arduino board and the smartphone. The motor choice will be based on the ratio of energy consumption/reliability/monetary cost and in the fact that piezoelectric based motors will need to be custom made for the application (making the question "Is it worth it?" really important).

Tying all up is the idea of this project being as cheap as possible. This will define every move of the development process together with the idea of robustness and relative reliability.

Since the full implementation is too ambitious for only one person and in such short time, the project is divided in two parts: the image processing part and the mechanic and control part. The image part is not a subject of this specific thesis. The main emphasis of this project will be the physical design of the application.

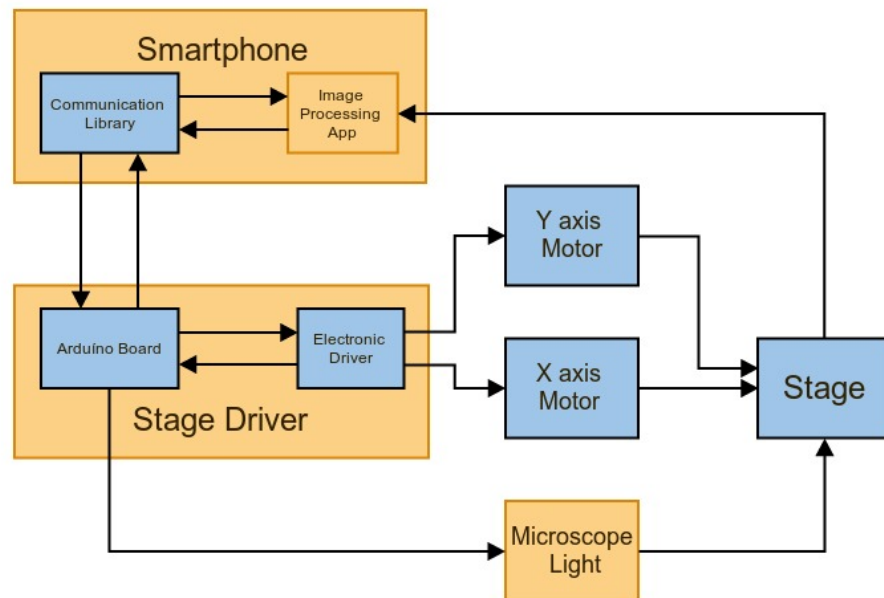


Figure 1.2: Block diagram of the concept.

Figure 1.2 represents the block diagram of the concept system. All blocks in blue are the sub-systems that will be developed by this thesis. The result will be integrated in the ICT4D MalariaScope [14] project, as said above, that aims to develop a full automated solution to diagnose malaria disease.

1.2.1 Requirements, Guidelines and Limitations

Although some requirements and guidelines were presented in previous points, in this section they are summarised for a better organization. Joining the requirements, some limitations need to be considered too.

- **Requirements:**

- The stage needs to move, at least, $20mm$ in each axis with a resolution of $300/400\mu m$;
- If possible, smartphone's battery must be the only energy source;
- The final product needs to operate under adverse conditions (high temperature, dust, non-technical operators, etc.);
- The design needs to be as cheap as possible and easy to replicate;
- Derived from the point above, the replication must be possible in third world countries;
- The smartphone's camera needs $2s$ to take the picture. The stage will need to stay in the same position for that time;

• **Guidelines:**

- The smartphone available for the project is a Samsung Galaxy S4 Zoom;
- All blood smears analysed have a maximum diameter of $2cm$ (assuming a circular form);
- System's autonomy needs to be acceptable ($\sim$ one hour minimum). If possible, an autonomous way to charge the battery should be designed;
- Maintenance should be easy to perform;
- A 3D printer (Makerbot Replicator 2x) is available for the project. If possible, all physical parts of the stage should be created using it;
- Use of screws should be avoided;
- The use of open/free software should be favoured. Proprietary variants are limited to those available in the company/faculty.

Presented requirements must be fulfilled without any exception. They are the backbone of the project and can't be overlooked. The guidelines are additions that could add value to the project and, if possible, should be addressed.

Some logistic and technological limitations need to be taken into account in the development. They can be checked below.

• **Limitations:**

- Budget available for material is around 300€ without the smartphone;
- Battery supplies $500mA$ at $5V$ (maximum, USB). Its capacity will be limited by the smartphone used;
- From the point above, Arduino board, electronic driver and any other components consumption must not exceed $500mA$ in any point;
- The 3D printer has a theoretical maximum resolution of $0.25mm$.

With this subsection, a better overview of the project can be created, leading to a better understanding of what is expected. The objectives proposed are discussed in next point.

1.3 Objectives Proposed

In this section are summed up the project objectives that will work as guidelines along the development:

- Elaborate a state of art report about:
 - micro-stage movement;
 - piezoelectric technology and understand actuators options;
 - electric motors technology and understand motor options;
 - Arduino/smartphone communication options;
- Choose the motor option and design it;
- Sketch, simulate and implement the motor driver controlled by an Arduino board;
- Assemble and validate the motor;
- Validate the possibility of using the smartphone's battery for all energy related tasks;
- Design and assemble the stage;
- Develop and fully test the Arduino/smartphone communication library;
- Fully test and validate the final system;
- Write the document detailing all the work done.

1.4 Thesis structure

This document tries to detail all the work done supported by a strong theoretical base and in a way that is understandable for the reader. The order is not chronological but logical: all the items are presented in a way that improve the flow of the reading and understanding:

- **Chapter 1** is self explanatory and presents all the information needed to understand the objectives of the project;
- **Chapter 2** presents a state of art review were almost all the actual solutions for the XY Stages are presented as a way to show the reader the study made before the project development. There is a study too about the actuator choice with an analysis of the pros and cons of each other;
- **Chapter 3** presents all the development made relating to the electronic driver used to interface with the actuators;

- **Chapter 4** focus in the development of the actuation system and the stage, showing all the phases surpassed until the creation of the actual prototypes. There is a subsection too detailing all the experience of working with the 3D printer focusing in modelling and calibration;
- **Chapter 5** relates to the development of the Android library used to build the example controller application. It focus too in the communications part of the firmware of the Arduino board. Features all the necessary information to use the library effectively (Javadoc and UML analysis);
- Finally, **Chapter 6** is self explanatory and features the conclusion part of the document. Exposes a set of conclusions and remarks of the work done, criticising it and evaluating the viability of the prototypes in terms of the objectives and the economic aspects. Presents a subsection too where all the future work is discussed and possible solutions presented.

Chapter 2

State of Art

In this chapter, we will start with the state of art analysis of all subjects relevant for the project. It will start with an introduction to xy-stages and their movement, then focusing in solutions adopted in literature explaining their actuator choice, topology, results and conclusions taken for the work in progress. After that, the main actuation options will be presented, explaining their technology, methods of control, pros/cons and limitations. As a direct consequence of having multiple options, they will be put against each other, comparing their advantages and disadvantages. A preliminary decision will be made after the comparison and a detailed explanation of the reasons will follow. To help the future work, an analysis of different mechanical systems used in products that have more or less the same objectives will be presented too. After this, the Arduino/Smartphone serial communication library will be presented.

The more careful reader might find strange the use of future in the text but that is a direct consequence of this preliminary work done before the development started.

2.1 XY-Stages

As years go by, the need of precision machinery rises in numerous fields like metrology [15], biomedical research [16] and semi-conductor manufacture [17]. For each of those fields, the needs change but one becomes really common: the increased need of precision movement of a determined point or object. To answer that need, one concept was created: the *XY-Stage*. But, before starting to analyse the literature, it would be very useful to answer the next question:

2.1.1 What are XY-Stages?

Well, different definitions can be presented. But all of them are well summed up in this quote directly taken from a respectable electronics corporation glossary:

XY-Stage — apply to motion as in a system; generally referred to as X, XY, and XYZ. The basic function of a stage is to constrain motion to a defined direction. For a linear stage, this motion is, ideally, along a straight line.

Motorized linear stages are used to precisely position objects along a single axis. They are comprised of a moving platform and stationary base joined by a bearing system. A motion controller is utilized to control position electronically. Motorized stages are particularly suited for high volume/high speed precision applications. [18]

This quote reproduces faithfully the concept of a XY-Stage. As a visual aid, the Figure 2.1 shows a state-of-art XY-Stage produced by Physik Instrumente [7]. The simplicity of the xy movement allied with the need of precision makes XY-Stages a powerful engineering solution for a good number of problems faced in everyday industrial applications as shown next.



Figure 2.1: M-545 XY-Stage from Physik Instrumente.

2.1.2 Implementing XY-Stages

Without any restrictions beyond respecting the definition above, a search was made to find out which solutions were created regarding the stage's movement. Various topologies can be found that are based in piezoelectric actuators, several electric motor types and electromechanical actuators. This seems to be the most used ways to develop a precision stage and some examples are worth further analysis.

2.1.2.1 Piezoelectric Stack actuator driven XY Stage

First off all, a quick explanation of what is a piezoelectric stack is needed to fully understand the example. A PZT Stack is a monolithic ceramic construction of many thin piezo-ceramic layers which are connected in parallel electrically. Applying a high voltage ($\sim 200V$ for maximum displacement) differential at its terminals generates a mechanical displacement (in the order of a few hundred μm) [19]. Figure 2.2 illustrates the actuator.

The PZT subject will be explored later in the document (see Section 2.2). For now this is enough to understand the topology. The example's stage design is aimed to Micro/Nano-manufacturing and has two axis. Each one is composed by a doubly clamped beam and a parallelogram hybrid flexure with compliant beams and circular flexure hinges. The stage is actuated



Figure 2.2: PZT Stack actuator [1].

by piezoelectric stack actuators, and two capacitive gauges were added to build a closed-loop positioning system [1]. The final design can be appreciated in Figure 2.3.

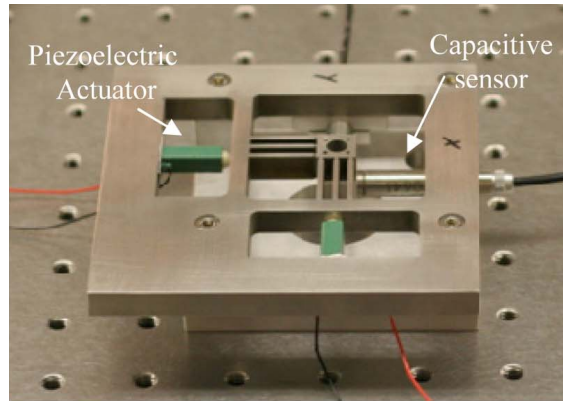


Figure 2.3: PZT Stack driven stage topology [1].

This design uses two independent kinematic chains that connects the actuator to the stage in parallel. Each kinematic chain includes a prismatic joint that is actuated by the PZT stack and a parallelogram hybrid flexure mechanism. This parallel connection almost decouples both axis (Figure 2.4).

This topology uses a DSP controller to generate the control signals. It implements a discrete PI controller (with feedback coming from the two capacitive gauges) with a sampling frequency of $20kHz$. The driver used is a commercial piezoamplifier with the maximum current of $1A$ and output voltage of $-30V$ to $150V$.

With this configuration, it is capable of about $15\mu m$ of motion along each axis with a resolution of about $1nm$. A test of this claim is presented in Figure 2.5.

After all the technical details, a pros and cons analysis must be made. The table 2.1 shows this information in a clear and informative way.

This design presents an impressive precision. The use of the capacitive sensors makes a strong difference as shown in Figure 2.5 but it increases production price. That precision comes to a cost

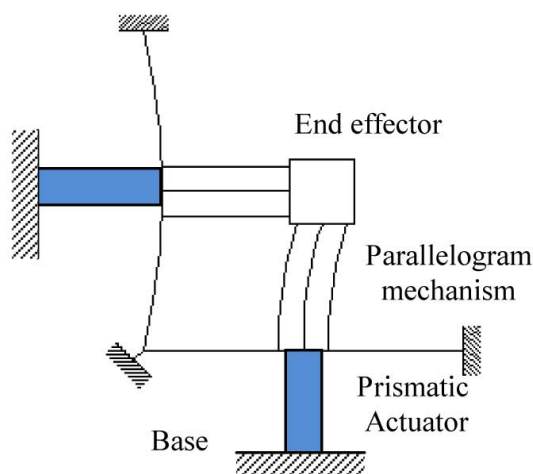


Figure 2.4: PZT Stack driven stage hinges [1].

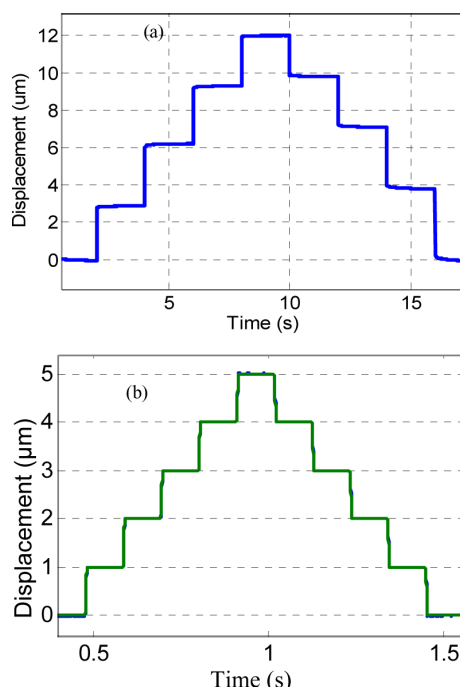


Figure 2.5: Open loop (a) and closed-loop (b) response to step motion [1].

in range of motion. $15\mu\text{m}$ per axis is a very small dimension and makes this design only useful for really specific applications.

The decoupling of the axis ensures that there is almost no parasitic theta movement. To accomplish that, a complex and precise hinge is needed, making it difficult and very expensive to produce.

Using a PZT stack actuator is a big plus. Its precision and very low power consumption (see Chapter 2.2) makes it a very attractive solution. Although depending on the distributor, it can be

Pros	Cons
Precision, $1\eta m$ resolution	Small range of motion, only $15\mu m$ per axis
Decoupling of each axis movement	Complicated kinematic chain
PZT Actuator	Commercial driver used
Closed loop control	Cost of the system

Table 2.1: PZT Stack driven stage pros and cons.

really expensive. The cheaper ones start from \$60 and the most expensive can go up to thousands of dollars (Prices by PiezoDrive [20]). Joining a commercial drive, the cost can go really high really fast.

Looking now for the objectives of this specific thesis, some ideas can be taken from this topology:

- The PZT actuator is really precise. Using one might be a great solution if one cheaper actuator alternative can be found. An amplifying system must be developed if the actuator is used directly;
- If more precision is needed, a closed-loop PI controller can be used to drive the actuators;
- Might be a good idea to decouple the axis. This can bring gains in terms of actuation velocity and precision;
- Precision is equal to complexity. Since the thesis objectives don't require maximum precision, it is a good idea to keep the design simple.

2.1.2.2 Piezoelectric Worm Actuator driven XY Stage

As in the previous example, a presentation of a PZT worm actuator is the first focus point. It is composed by three extensive PZT actuators (stacks for example) that use the concept of clamping as illustrated in Figure 2.6. The control is achieved by applying a voltage differential to each actuator.

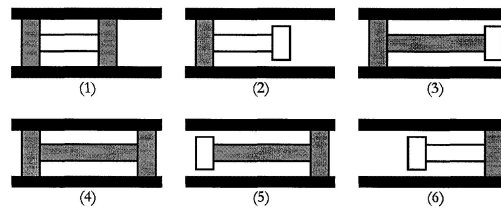


Figure 2.6: PZT Worm actuator step sequence [2].

This example is aimed at generic micro-applications and uses a modified worm actuator. A piezoelectric component was mixed with mechanical clamps as a way to achieve simplicity and

increased force. One of them is naturally clamped (with low voltage) and the other one is the opposite. The complimentary clamps make possible to drive the motor with a 2-channel controller, using one channel for the clamps and another for the piezoelectric actuator [2, Piezoworm Motor Designs].

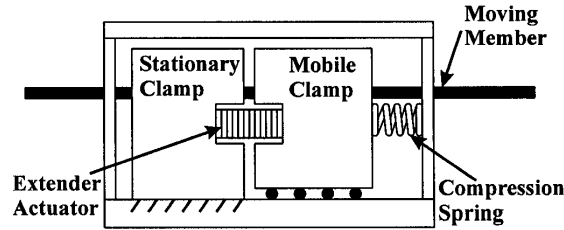


Figure 2.7: PZT Worm Motor concept [2].

The motor actuation drags the stage along a supporter slide. To achieve XY movement, two of this systems were used. One of the them moves with the stage in order to achieve independent actuation.

To drive the motors, a computer with Labview was used. It received feedback from 2 encoders that were placed in each motor implementing a Two-Stage controller. The first stage is the Regulatory control that generates the signal for the piezoelectric component. The second stage is the Supervisory control and generates the clamp's signal [2].

A commercial electronic driver was used (DSM VF-500 [21]) and it was capable of supplying 200V and 500mA.

This controller, along with the feedback, achieves performance with a maximum error of $10\eta m$, displacement of 10mm in each axis with a sample frequency 10kHz and a speed of 8.5mm/s. Results can be observed in Figure 2.8.

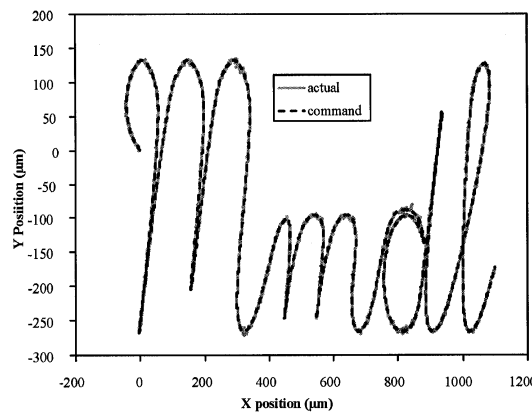


Figure 2.8: PZT Worm driven stage example results [2].

Pros and cons analysis for this implementation can be observed in Table 2.2.

Pros	Cons
Good precision, $10\eta m$ resolution	Use of mechanical clamps
High displacement, $10mm$ in each axis	Commercial electronic driver
Simple but effective design of the actuator	Use of a proprietary software running in a commercial PC
High actuation force (maximum of $\sim 35N$ [2, Image 3.8])	Use of encoders (limited resolution)
	Expensive

Table 2.2: PZT Worm driven stage pros and cons.

Some good ideas were presented in this work. The actuator topology is very ingenious surpassing the natural low displacement of the PZT stack and turning the stage into a virtually infinite displacement system. This capacity comes with a cost of using mechanical parts in the design that are subject to wear and need maintenance to keep the performance. That wear can be a problem for the control mechanism because the system response will change over time. In favour of the clamps is the fact that they provide an increased force to the actuator.

The use of the commercial driver is a big con. Those drivers have high performance but are really expensive. If the cost is not a concern, those should be used. Otherwise, custom drivers give a lot more flexibility because they can be built to meet the requirements of the application without having excess functions.

Another big con is the use of encoders. Cheaper ones don't have a good resolution and since the control is based on their signal, if a good resolution is needed and that will not be cheap.

Finally, the use of a commercial PC and software is a really weak solution. Processing needs are not critical in this project because the quantity of data to be processed is low (2 encoders and some simple PI calculations). A dedicated solution (DSP, micro-controller, etc.) was a better fit for the project.

As shown, money was not a concern in this example so, some solutions might be impractical if we take it into account. Nevertheless some ideas are really relevant for the thesis in progress:

- Again, PZT actuators were chosen to build the mechanism of movement. This seems to be the rule in XY-Stages;
- Low displacement (and force if it is an issue) of the PZT actuators can be surpassed with good imagination and creativity;
- Use of commercial electronic drivers is really common too. Might be a good idea to make a market survey;
- Closed-loop control can be achieved with different sensors. But using one limits the system resolution to its precision.

2.1.2.3 Comb-drive Actuator driven XY Stage

This example is the first presented that an PZT actuator is not chosen to drive the stage. A electrostatic comb-drive is used but its principle is not very different. Figure 2.9 represents a schematic of the presented actuator. A comb-drive actuator comprises two electrically isolated conductive combs with a set of fingers equally spaced. One of the combs is moving and is guided by a flexure mechanism with really low stiffness. The other is the opposite (fixed, high stiffness).

As said above, the actuation principle is really close to PZT actuators. When a voltage differential is applied between the combs, an electrostatic force is generated and the moving comb is pulled to the fixed comb along the Y axis. A parasitic movement is induced in X axis that is compensated by the flexure mechanism. Maximum displacement is limited by the finger width and by the flexure mechanisms [3].

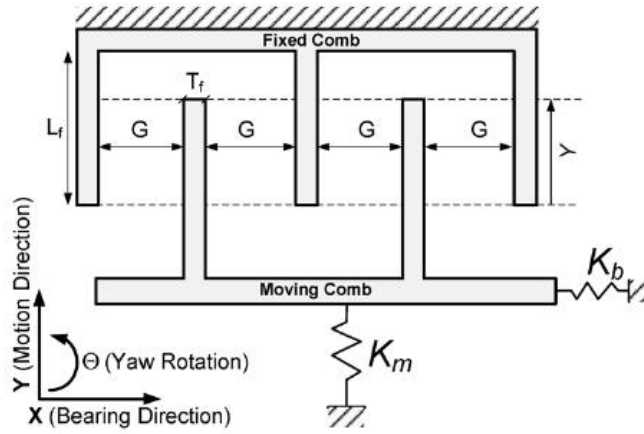


Figure 2.9: Electrostatic Comb-drive actuator [3].

This stage (aimed at generic micro-applications) uses an altered comb-drive actuator in each axis. It is composed by two comb-sets to provide bi-directional actuation capability. To achieve total decoupling, intermediate stages were used as shown in Figure 2.10. In the proposed topology, the comb-drives are placed between ground and their intermediate stage to provide guidance between both combs of the actuator [3].

To drive the actuators, a commercial controller and electronic driver is used (a Keithley 4200-SCS). It can provide up to 210V and 1A. This is a test suite, not a final implementation of the controller and no feedback is used.

With this topology, a displacement of $228\eta m$ is achieved with a resolution of $1\eta m$. An image of the results can be observed in Figure 2.11.

Table 2.3 gives an overview of the pros and cons analysis.

The solution developed in this example features high resolution but a low displacement. The classification of the displacement as low is subjective because the thesis in progress is used as term of comparison but can be really high depending the specific application.

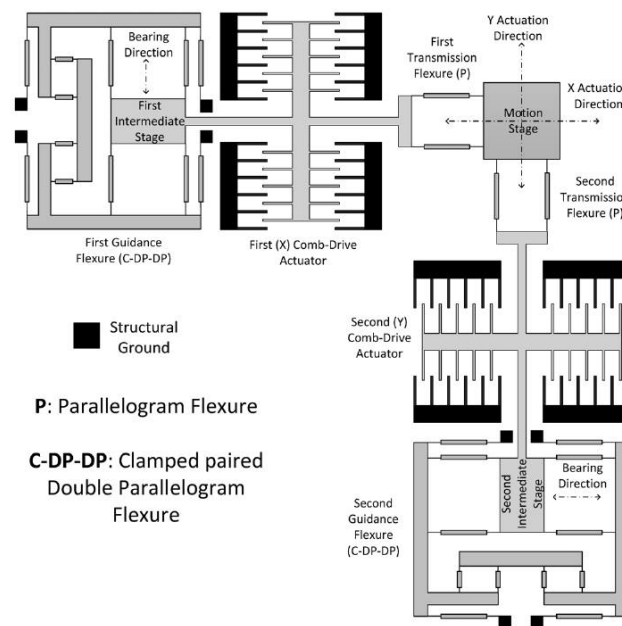


Figure 2.10: Electrostatic Comb actuator driven stage topology [3].

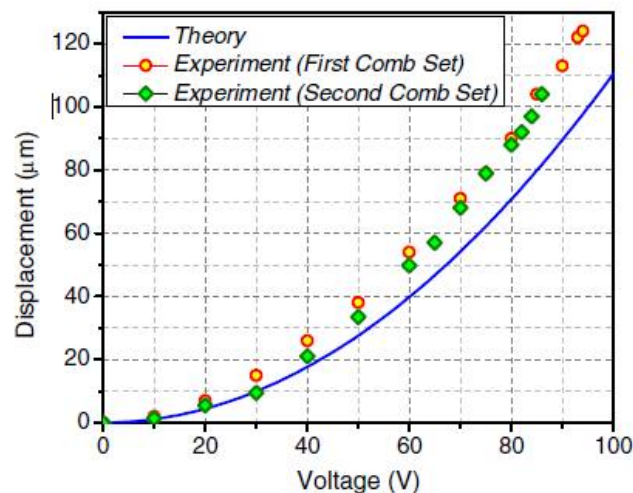


Figure 2.11: Electrostatic Comb actuator driven stage results [3].

Since this was clearly a work in progress, the solution for the controller and driver should be despised. What was used was only a testbench machine, not representative of a developed solution.

The actuator has a simple principle but its application is difficult. A high precision in the development is needed because the flexure mechanisms are the bottleneck in terms of force and repeatability and that's a big con.

Tying everything up, is the cost. No specific pricing was found but the need of intermediary stages and mechanical components joined by the comb actuator price, makes this a costly solution.

Pros	Cons
High precision, $1\eta m$ resolution	Complex design, high precision needed in implementation
Decoupling of each axis movement	Commercial electronic driver and controller
	Low displacement, $228\eta m$
	Low force, limited by mechanical components
	Expensive

Table 2.3: Comb Actuator driven XY stage pros and cons.

Relating the previous information with the work in progress, we reach the following conclusions:

- Electromechanical actuators exist and can be used for this project. A trade-off exists between using them and increasing complexity due to the mechanical components;
- Using mechanical parts in the design turns it more expensive. They should be avoided if possible.

2.1.2.4 Stepper Motor driven XY Stage

In the examples shown so far, only PZT and electromechanical actuators were used. In a more traditional way, this example uses simple circular stepper motors. This was not found in published work but in a hobbyist website [5].

First, a brief overview of stepper motors is in order. It uses electromagnetic properties to convert digital pulses into mechanical rotation [4]. It is composed by windings that, when energized in sequence, provide movement in steps whose size is dependent of the physical components.

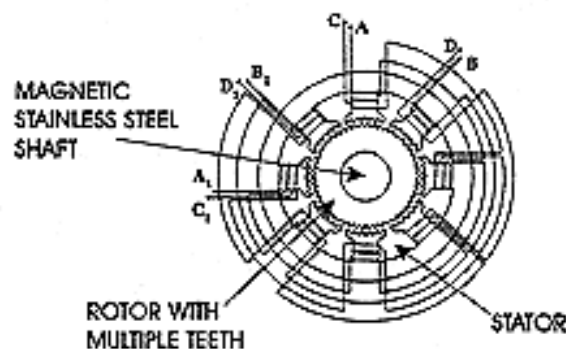


Figure 2.12: Stepper Motor [4].

A more detailed insight of the stepper technology can be found in Section 2.3. Moving to the topology, this example represents a microwell plate [22] that features an enthusiast build with really cheap components. This is interesting because all other examples found in literature feature expensive designs and are focused in precision. The idea here is to get the work done as cheap as possible.

The design features two unipolar stepper motors to provide fully decoupled movement in each axis. The stage is guided by stainless steel rods along with linear ball bearing holders. To move the stage, a toothed belt is used along with sprockets [5].

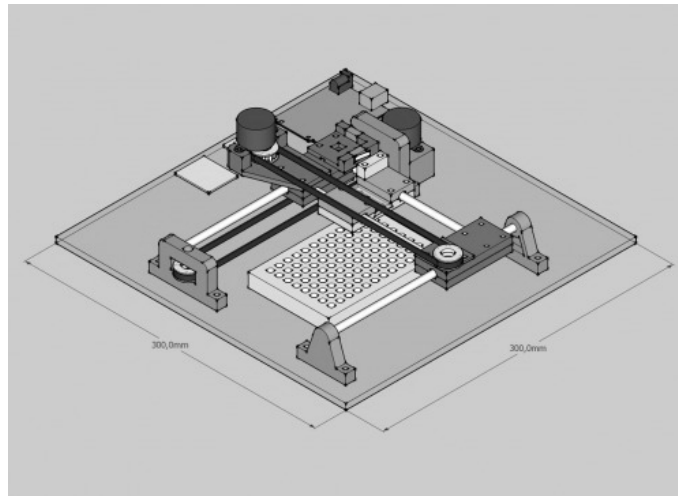


Figure 2.13: Stepper Motor driven stage topology [5].

An Arduino [23] is used as a controller. To drive the motors, a simple circuit based in an ULN2003A chip (is an array of transistors) is used and everything is powered by the Arduino (USB connection, maximum of 500mA).

No information of the resolution is presented but it will be roughly the same of the stepper motor with a bit of error inserted by the mechanical parts and friction. The step size is limited too by the motor although an algorithm of micro-stepping [4] can be used that divides each step in micro-steps at cost of repeatability. In the example, the motor used rotates 360° in 4096 steps. To traduce that in metric dimensions, the mechanical belt and the sprocket specifications must be known and no information is given.

The design completes a run of a twelve by eight cm grid in less of four minutes. A video of its functionality can be observed in the project website.

A pros and cons analysis of this design can be observed in Table 2.4.

This example features a cheap, fully working stage with a large travel range and simple control. The use of a 3D printer is a really nice solution as shown because everything can be customized and fit every requirement. The use of sprockets and toothed components can be a problem if a more precise design is need.

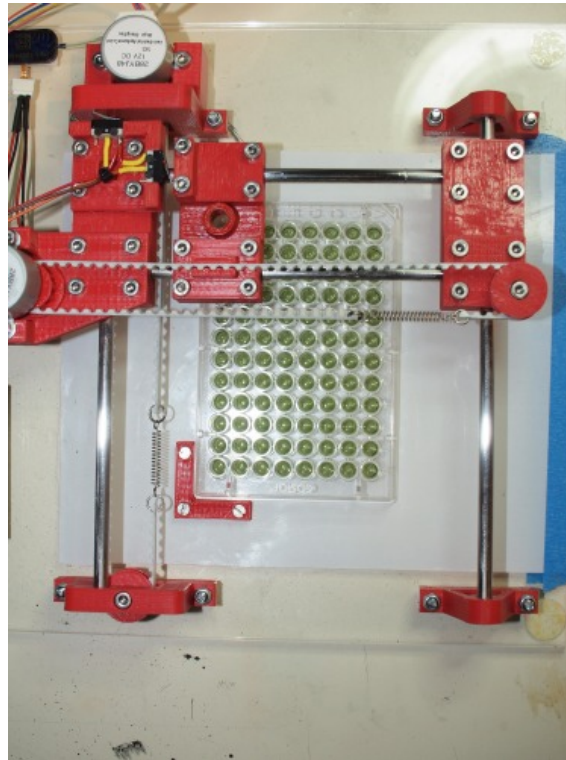


Figure 2.14: Stepper Motor driven stage results [5].

Pros	Cons
Simple design	Low resolution (can be surpassed)
Really cheap project ($\sim \$100$)	Use of mechanical components (sprockets, screws, toothed belt)
Structure 3D printed	Slightly slow
Simple controller and electronic driver (Arduino based)	
Very high displacement, 120x80mm	

Table 2.4: Stepper Motor driven XY stage pros and cons.

This is a very good solution for some applications but the bottleneck is in the motor and resolution of the 3D printer. Looking now to the thesis objectives, a lot of good solutions and remarks can be achieved:

- The use of a 3D printer improves development time and reduces project cost. Preferably, it should be used;
- Stepper motors can be a solution if enough resolution can be achieved. The use of complementary mechanical parts can be a setback;

- A controller based in Arduino is really powerful and cheap. This example shows that it was a good choice for the thesis;
- If the actuator used consumes little power, a electronic driver powered by the Arduino can be used;
- Closed loop control is not needed for good performance.

2.1.3 Remarks

Some examples were presented in the subsections above that were representative of some ideas and solutions which can be used in the project in hands. As shown, PZT actuators are the preferred to drive the stage because of its high resolution and low power consumption. But they are limited in terms of displacement and for the thesis that's a big con. Electromechanical actuators are used too but they are excluded because they bring unnecessary complexity, high cost and maintenance. The other valid actuator found in literature was the stepper motor that is really simple and cheap but with limited resolution. The fact that the electronic driver can be implemented using only the power from the controller (Arduino) is really relevant.

In Table 2.5 a summary of all topologies is presented showing the relevant points of each one as a way to choose the actuators that are worth a deeper study.

Analysing the table, one can see that if a intermediate solution with the characteristics of the PZT actuators and the cost of the stepper motors can be found, it will be one of the best solutions available. So, a deeper analysis of the PZT technology is needed to address this possibility and it can be found in Section 2.2.

As a backup plan, a solution based in stepper motors is really relevant. Its characteristics make it a valid solution if paired with a good imagination to overcome the low resolution problem. A analysis of this technology is presented in Section 2.3.

Regarding other implementations and topologies, every written work studied presented solutions based in electromechanical actuators like the one shown in Subsection 2.1.2.3. The option to present only one example was taken because every one had the same cons i.e. really high difficulty of implementation (as a result of it, very high price too), expensive drivers, high maintenance needed, etc. (see Table 2.5) and that violates the principles of the thesis. Because of this fact, electromechanical actuators are excluded for now (as said above) and considered not suitable for a in depth analysis leaving the battle for the PZT and Stepper technologies.

Topology	Pros	Cons
PZT Stack	Precision, $1\eta m$ resolution	Small range of motion, only $15\mu m$ per axis
	Decoupling of each axis movement	Complicated kinematic chain
	PZT Actuator	Commercial driver used
	Closed loop control	Cost of the system
PZT Worm	Good precision, $10\eta m$ resolution	Use of mechanical clamps
	High displacement, $10mm$ in each axis	Commercial electronic driver
	Simple but effective design of the actuator	Use of a proprietary software running in a commercial PC
	High actuation force (maximum of $\sim 35N$)	Use of encoders (limited resolution)
		Expensive
Comb-drive Actuator	High precision, $1\eta m$ resolution	Complex design, high precision needed in implementation
	Decoupling of each axis movement	Commercial electronic driver and controller
		Low displacement, $228\eta m$
		Low force, limited by mechanical components
		Expensive
Stepper Motor	Simple design	Low resolution (can be surpassed)
	Really cheap project ($\sim \$100$)	Use of mechanical components (sprockets, screws, toothed belt)
	Structure 3D printed	Slightly slow
	Simple controller and electronic driver (Arduíno based)	
	Very high displacement, $120 \times 80mm$	

Table 2.5: Summary of all topologies shown.

2.2 Piezoelectric Technology

This section discusses piezoelectricity and piezoelectric actuators. It starts by referring briefly its history and explaining the principles behind the technology, then moving to a full review on piezoelectric actuators and their topologies, addressing next how to drive them and then finishes with a subsection dedicated to remarks and conclusions. A brief introduction was made in Subsection 2.1.2.1 and in Subsection 2.1.2.2 and here is presented a more detailed analysis.

2.2.1 Historic Review

The name "piezo" derives from the Greek, meaning "to press". This effect was discovered by Curie brothers when they found that certain materials (topaz, quartz, etc.), when under mechanical stress, developed macroscopic polarization, leading to electrical current (PZT effect) [6]. Later, in the following year, Lippmann predicted the inverse PZT effect, the application of an electric field to a PZT crystal leads to a physical deformation of the said crystal and it was proved by the Curie brothers latter in that year [24].

Since then, this technology was used in crystal resonators, sonar's, microphones, piezo-ignition systems, accelerometers, ultrasonic transducers and even electronic chips [25].

2.2.2 Piezoelectric Effect

First, a definition of a dielectric material is need. It is an electrical insulator that can be polarized with an electric field. The electric charges do not flow like in a conductor, they just slightly shift from their position and align in the field direction [26].

A piezoelectric material is a class of dielectric materials that can be polarized not only by an electric field but also with application of an external mechanical force.

Figure 2.15 sums the interactions in a simple way.

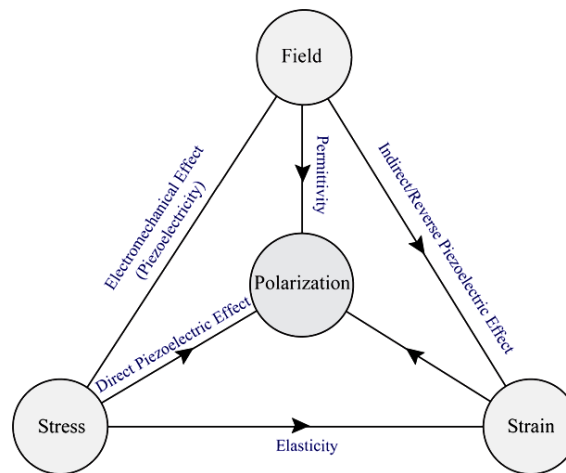


Figure 2.15: Piezoelectric interactions [6].

The piezoelectric effect corresponds to the generation of an electric charge as a result of a force exerted in the material. Before applying the external force, the negative and positive charges of each molecule coincide resulting in a neutral particle. When the force is applied, the molecule can be deformed, forming little dipoles. Inside the material, the poles cancel each other and fixed charges appear in the surface. This means that the material is polarized and represents the direct piezoelectric effect [6].

The reverse piezoelectric effect is observed too in some materials i.e. applying a voltage in the electrodes produces a mechanical deformation in the actuator. This way of transforming

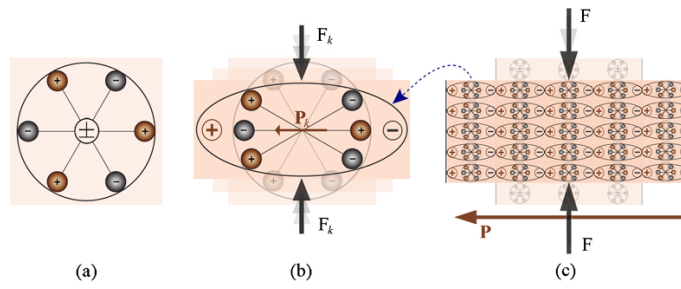


Figure 2.16: Direct piezoelectric effect. a) Normal molecule. b) Formation of a dipole. c) Effect of an external force in the material [6].

the electrical energy into usable mechanical energy is fundamental to the applications like those presented prior in this document.

When applying this principle, one must take in account that the material suffers from hysteresis. When a voltage is applied to the material, the polarization follow the curves (i) and (ii) on the P-E plot of Figure 2.17. When the voltage is removed, the polarization follows curves (ii) and (iii). As can be seen in the plot, some polarization is retained when the voltage is removed making the curves different [6].

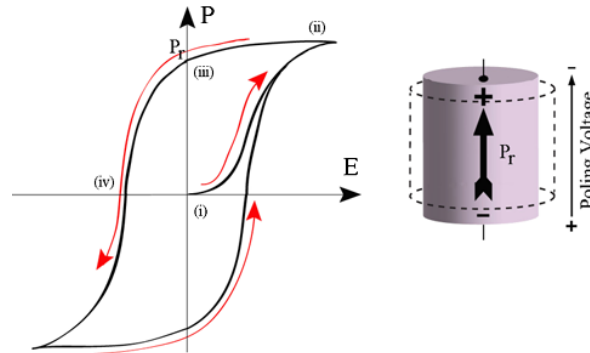


Figure 2.17: PZT polarization curve [6].

Ignoring the hysteresis, the relationship between the displacement and electric field strength for a single PZT element can be expressed as $\frac{\Delta l}{l} = dE$ where the constant d is the piezoelectric strain constant, E is the electric field and $\frac{\Delta l}{l}$ is the strain. Because $E = \frac{U}{l}$, then $\Delta l = dU$ and this concludes that the displacement (Δl) is independent of the dimensions and dependent of the voltage applied (in V) [7].

As shown, PZT elements have a direct relationship between displacement and force. If more force is needed, less displacement the actuator will have and the other way around. So, the producers objective is to find the best trade-off between both properties using plots like the one shown in Figure 2.18.

Other property of this effect is the low consumption of the material. Basically, the power

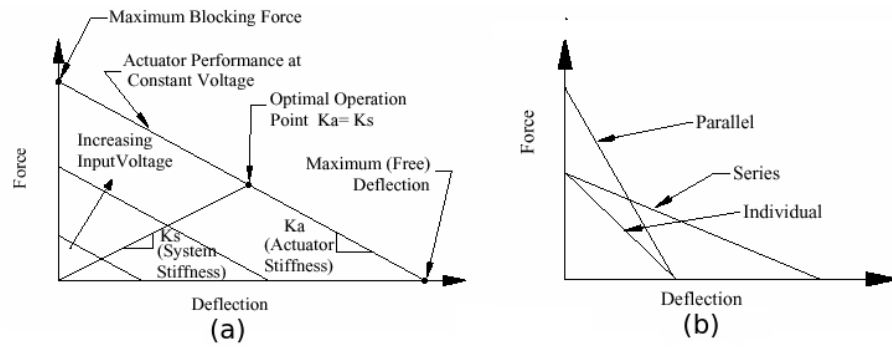


Figure 2.18: Force versus Deflection of PZT actuators. a) Individual actuator analysis. b) Different behaviours of the 3 possible assemblies [7].

needed is just the one to polarize the molecules. The equivalent circuit of a PZT actuator is illustrated in Figure 2.19 and it simplifies to the charge of an capacitor [27]. The capacity can be estimated by $C = n\epsilon \frac{A}{d}$ where n is the number of layers ($\frac{l}{d}$, l is the length and d the layer thickness), ϵ is the dielectric constant and A is the area of a layer [7].

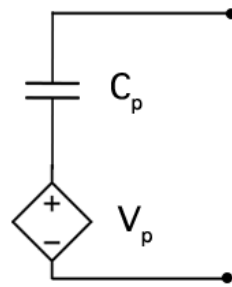


Figure 2.19: PZT actuator equivalent circuit [7].

2.2.3 Basic Piezoelectric Actuators

After the discussion about the technology principle, the most basic actuators based on it are presented. All PZT solutions developed have in its base two simple actuators: the PZT stack or the PZT bender. They are the simplest form of application of the technology and are composed by several thin slices of piezo-ceramics put together with high pressure [19]. Both actuators use the same principles but in different ways and that is discussed next.

2.2.3.1 PZT Stack Actuator

This basic actuator uses the premise presented before. It uses several thin slices of piezo-ceramic on top of each other in a way to achieve more linear displacement while maintaining a low

voltage requirement. It features a typical deformation of a several dozen of μm and a typical force of several hundred N . The cheapest example found costs around \$60 (Prices by PiezoDrive [20]) making it a somewhat expensive solution.

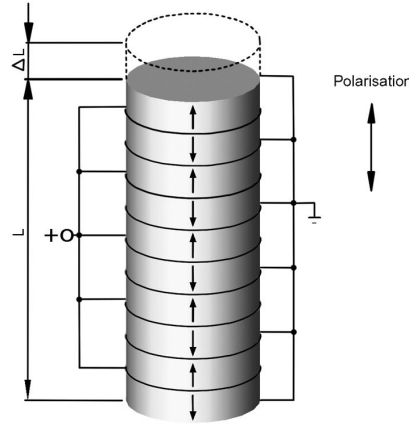


Figure 2.20: PZT stack actuator by Pi Ceramic.

The stroke of the actuator can be calculated using $l_{stroke} = NdU$ and its blocking force with $F = \frac{l_{stroke}A}{NtS}$ where N is the number of piezo-ceramic layers, A is the section (m^2) of the actuator, t is the layer thickness and S the elastic constant ($\frac{m^2}{N}$) [19]. The variables not mentioned are defined in 2.2.2.

Since they have very low displacement, this actuators are often used with amplifying systems and that will be discussed in Subsection 2.2.4.

2.2.3.2 PZT Bender Actuator

Contrary to the PZT stack, the bender actuator is, usually, composed by two piezo-ceramic plates (bimorph) which are joined together. During actuation, one of them is extended and the other is contracted. This principle generates the bending movement that is observed [19].

The bimorph bender actuator can be of two types and they are distinguished by their interface:

- **Serial:** has two electrodes (one per layer) where one is connected to ground and the other to the amplifier voltage;
- **Parallel:** features three electrodes (one per layer and one for a extra middle electrode). The surface ones are connected to ground and to a fixed voltage and the middle one is connected to the amplifier voltage.

This topology results in a bigger displacement (up to $\sim 2mm$) but a lower force ($\sim 1N$) when comparing to the previously presented stack actuator. The main pro of this actuator is the price, it can be found for \$15 (Prices by PiezoDrive [20]).

The stroke of the actuator can be calculated using $l_{stroke} = \frac{3T}{2L^2}dU$ and its blocking force with $F = \frac{3TW}{8L}YdU$, where T is the thickness, L the length and W is the width [19]. The variables not mentioned are defined in 2.2.2.

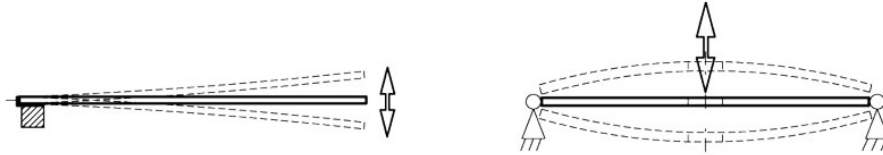


Figure 2.21: PZT bender actuator by Pi Ceramic.

A reference must be made to the unimorph topology that features only one piezo-ceramic layer but it isn't very common derived of its low displacement compared to the bimorph type.

Since this actuators have low force, imagination and creative thinking is needed if the actuator can't be directly applied. Some solutions are discussed in 2.2.4.

2.2.4 Piezoelectric Actuators and Motors

Here are discussed the typical arrangements used for applying the previous actuators to various situations. They can be summed up in the following categories [19]:

- **Mechanically amplified actuator** - since the PZT actuators have low displacement, a mechanical system based in levers or flexure hinges is used to improve it at cost of force - see 2.2.4.1;
- **Inertial-based actuator** - uses the friction and inertial forces to generate displacement - see 2.2.4.2;
- **Worm actuator** - uses stack actuators in a creative way, providing virtually infinite displacement but with an increased cost and limited force - see 2.2.4.3;
- **Ultrasonic motor** - similar to the worm actuator but works at its resonant frequency to provide larger output force - see 2.2.4.4.

Those categories are discussed in next subsections.

2.2.4.1 Mechanically amplified actuator

Since stack actuators provide high force but low displacement (the inverse for the bender), sometimes the direct application isn't easy and some auxiliary systems must be used. The simplest way to amplify the displacement is by using a mechanical lever. It needs to be rigid to provide good efficiency. The amplified output is proportional to the lever size with improved displacement and lower force (same proportion) [28].

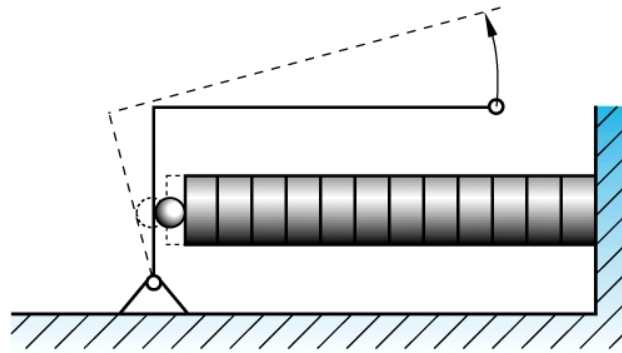


Figure 2.22: Simple PZT stack displacement amplifier.

Another way to amplify the displacement is using the diamond flexure system. It can be observed in Figure 2.23. This kind of actuator can produce high displacement, force and frequency. Usually, more than ten times the displacement can be achieved with this system.

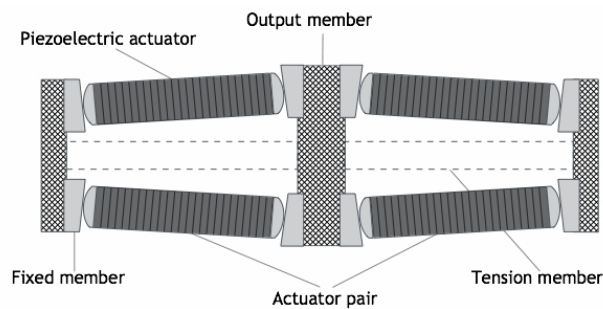


Figure 2.23: Diamond actuator system by Noliac [8].

An example of an actuator based on a PZT bender is shown in Figure 2.24. The hydraulic fluid is actuated by the bender actuator and drives a small piston cylinder. It is hydraulic connected to a larger one to produce the amplified displacement.

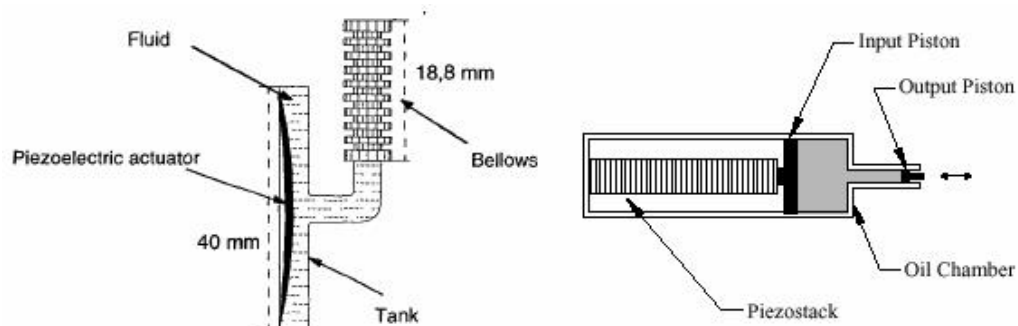


Figure 2.24: Hydraulic amplification system.

2.2.4.2 Inertial-based actuator

This method uses the connection between friction and inertial forces to produce the displacement. A good example is the stick and slip actuator shown in Figure 2.25. Each step movement consists in a slow deformation of the legs followed by an abrupt jump backward [28].

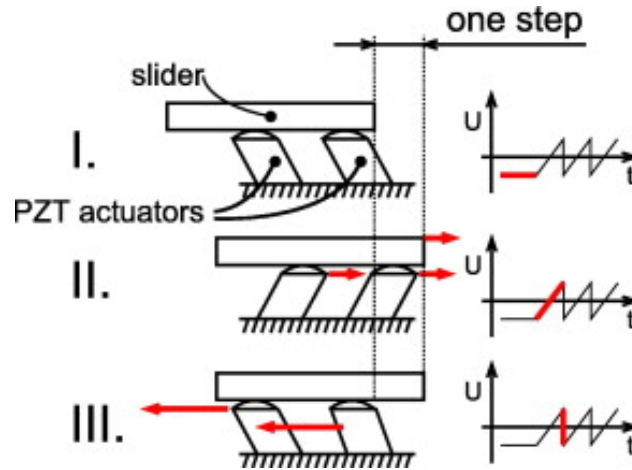


Figure 2.25: Stick and slip principle by PI [7].

Another example is shown in Figure 2.26. Here, the use of friction is fundamental. The force applied by the used point of contact (a rough material helps to improve the movement) is divided in x and y force. If $F_x > F_y$ and $F_x > P + F_a$ where P is the weight in N and F_a is the friction force (contrary to the movement), the system will move in little steps. A PZT bender or stack can be used as actuator.

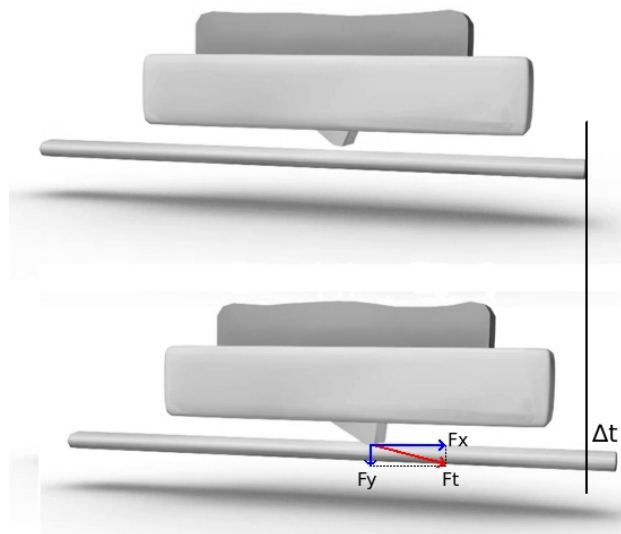


Figure 2.26: Friction slide principle.

2.2.4.3 Worm actuator

PZT worm actuators are composed by a set of three PZT stacks that work together to achieve theoretically infinite displacement. Two of the actuators work as clamps and the other one as an extensional element. In Figure 2.27 can be observed a typical step loop. First, one of the clamps contracts leaving the actuator supported by the other clamp (2). Then, the extensional element expands (3) and the clamp extends (4). Afterwards, the other clamp contracts (5) and the extensional element returns to its original size dragging the clamp (6) [2]. This cycle can be repeated as many times as needed.

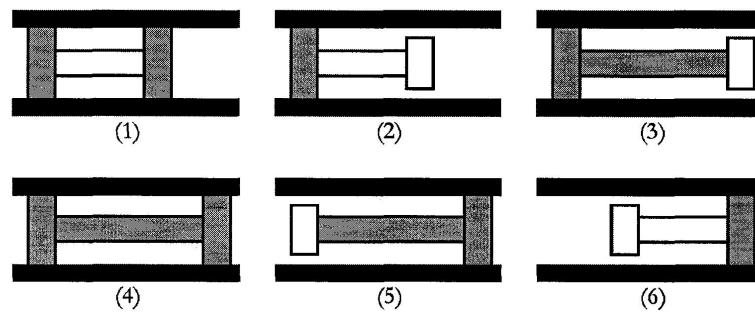


Figure 2.27: Worm motor actuation cycle.

2.2.4.4 Ultrasonic motor

Figure 2.28 shows an example of an ultrasonic linear PZT motor. The moving table is driven by bending actuators. They are excited with the same electrical source with same frequency but with a controlled phase difference. At the vibration tip, an elliptical motion is created. A vibration circuit working at the resonant frequency is used to cause the ceramic components to vibrate [19].

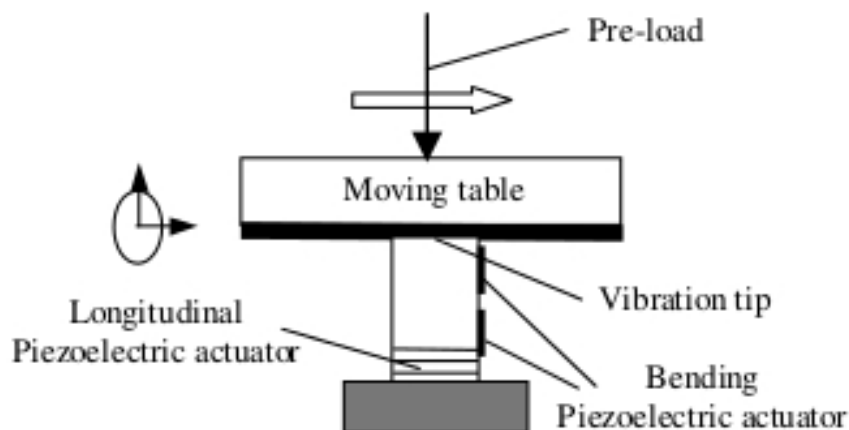


Figure 2.28: Ultrasonic motor example.

2.2.5 How to drive a Piezoelectric Actuator?

For this subsection, some guidelines must be defined. The best solution based in this technology is the PZT bending actuator as a step motor (see 2.2.6) and here only will be shown the driver topologies for it. For the other actuators, a full analysis can be found in [29] and [30]. Relating to the step motor, the control can be with constant tension because the frequency is not relevant (can have a delay in the order of seconds). Making use of the natural charge and discharge of the PZT actuator (it can be simplified by assuming it is a capacitor), only the maximum tensions need to be applied because a step cycle is composed by displacing the actuator to one side and then to the other. The discharge of the actuator guarantees that it passes by all positions. It is assumed that the bender actuator is rated at 100V.

Because of the different types of the bending actuator (two or three wires), two driver topologies must be developed.

The first type is the serial. As stated in 2.2.3.2, it features two wires. To achieve full displacement for each side a voltage of $-100V$ and $100V$ is needed. Since the available source is of 5V, the following driver can be sketched:

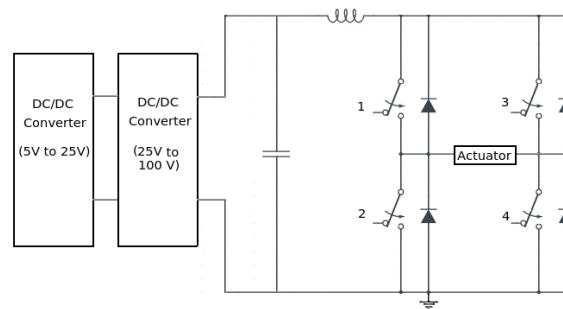


Figure 2.29: Power Driver for a serial PZT bender.

First thing that the driver needs is to be able to perform is to elevate the tension from 5V to 100V. To achieve this a double stage DC-DC converter is used and it can be observed in Figure 2.30. The micro-controller controls the output tension by adjusting the duty cycle of the switch. A staged topology is used because one converter won't be able to elevate directly to 100V. The capacitor represented in the full topology is the one from the second DC-DC converter. The coil is used to prevent big variations in the current while the transitions happen. Then a H-bridge is used. Adjusting its switches in opposite pairs applies $+100V$ (switches 1-4) or $-100V$ (switches 2-3) to the actuator. 0V can be applied if switches 2 and 4 are closed at the same time.

For the parallel one, a DC-DC converter is needed too. The difference is that in this case the tension needs to go to 200V because the outermost connectors need to be constantly at $+100V$ and $-100V$. Then, a half-bridge is used. If the switch 1 is on, the tension on the central conductor will be $+100V$. If the switch 2 is on, the tension will be $-100V$.

All capacitors need to have, at least, ten times the capacity of the actuator to ensure that they can charge it.

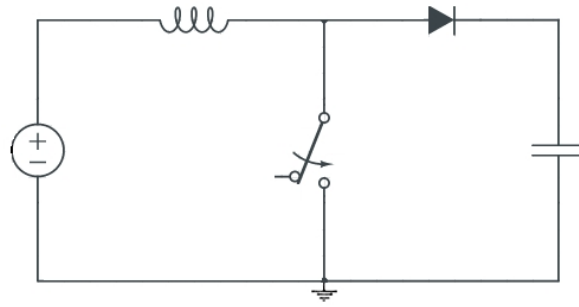


Figure 2.30: Step-up converter.

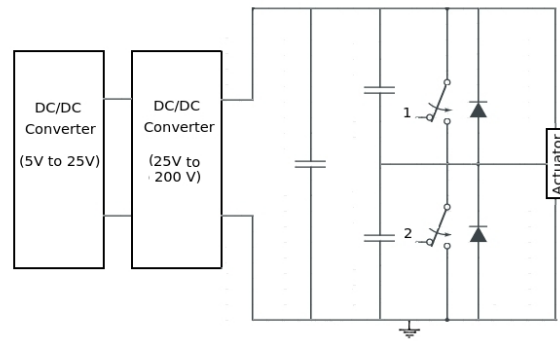


Figure 2.31: Power Driver for a parallel PZT bender.

2.2.6 Remarks

The properties of this actuators are really relevant for the thesis. Their low power consumption and easy drive allows an easy application. The big difficulty is to find a way to obtain displacement amplification. Some solutions were discussed in previous sections but the ones that seem more interesting go around the problem completely (worm or inertial) providing steps instead of trying to provide direct displacement. This is a really ingenious approach because it provides virtually limitless displacement but needs to be well executed, otherwise the resolution or even the complete functionality is in in danger.

If a PZT technology solution is to be used, the better option is clearly the PZT bending actuator as a step motor (friction-based slip motor). If well planed and developed, a good resolution and step size can be achieved with a really low cost (just the actuator for $\sim$ \$15 and the structure that can be produced in a 3D printer). Other options use expensive stack actuators and applying directly the bender doesn't provide enough displacement (needs to be higher than 10 mm, mechanical lever isn't an option because of the low force of the actuator).

2.3 Stepper Motor Technology

This section discusses stepper motors technology and starts by giving a brief overview of their history, explaining the principles behind the technology, then moving to a full review of the different motor topologies, addressing next how to drive them and then finishes with a subsection dedicated to remarks and conclusions. A brief introduction was made in [2.1.2.4](#) and here is presented a more detailed analysis.

2.3.1 Historic Review

Stepper motor technology history can be traced back to the 19th century with the name of Electromagnetic engines. Modern motors are due to Walker to whom was awarded a patent in 1919 by the invention of the tooth structure on both the stator and the rotor. This opened the field for the invention of the nowadays stepper motor by Thomas and Fleischauer in 1957 [\[31\]](#).

Since then, this motor concepts haven't changed much and they are used in several fields. The first industrial user was Fujitsu Fanuc in their world famous Fanuc series of CNC systems. As the technology was maturing, the size of the motors was diminishing and now we can find an exemplar in almost any printer. Examples of this technology are used in camera lenses, slot machines, scanners and compact discs drives.

2.3.2 Types and Operation

Related to construction, stepper motors can be divided in two major classes: with and without permanent magnets [\[31\]](#). In those classes we can find various topologies but the most representative ones can be observed in [Figure 2.32](#). Each class and its operation is described next.

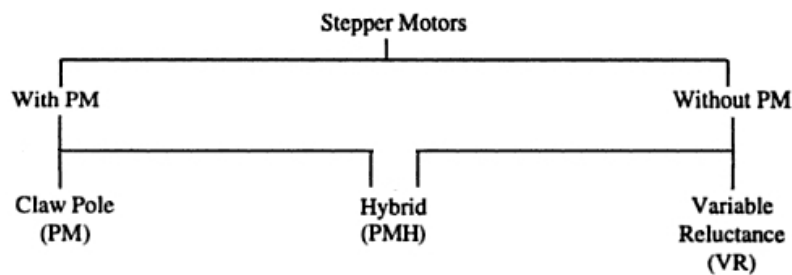


Figure 2.32: Stepper motors classes.

2.3.2.1 Variable Reluctance Stepper Motor

This type of motor does not use a permanent magnet. All the windings are located in the stator and the rotor moves without constraint. This solution is optimal for applications that don't require a high degree of motor torque.

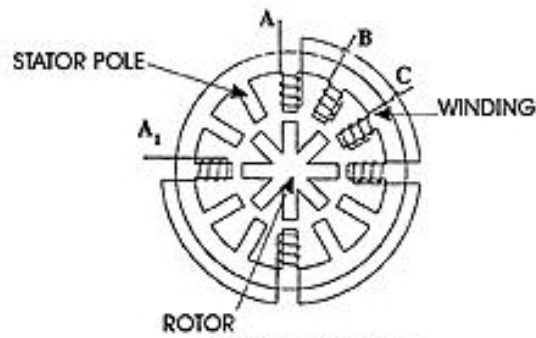


Figure 2.33: Variable Reluctance stepper motor.

The motor shown in Figure 2.33 has three points of winding energization (A, B, C) set 15° apart. Energizing pole A through the winding causes a magnetic attraction and the rotor teeth align with the teeth of the pole. Next, de-energizing pole A and energizing pole B causes the rotor to move 15° to align its teeth with the pole. Repeating this procedure to pole C and then restarting the cycle causes the clockwise movement. If the order of the cycle is reversed (A, then C and then B), it causes a counter-clockwise movement [31] [4].

2.3.2.2 Claw Poles Stepper Motor

This is a permanent magnet motor and is also referred as "canstack" motor [31]. It features a relatively low speed and torque with step angles of 45° or 90° . Its main advantage is its low manufacturing cost making it perfect for non industrial applications [4].

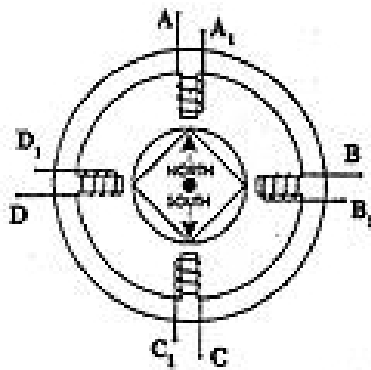


Figure 2.34: Claw Poles stepper motor.

The motor shown in Figure 2.34 has four phases. Unlike the previous example, applying current to each phase in sequence causes the rotor to adjust to the magnetic field (providing steps of 90°). This configuration features a relatively high torque and low speed.

2.3.2.3 Hybrid Stepper Motor

This is the most popular type of stepper motors, combining the best characteristics of variable reluctance and permanent magnet variations. They are constructed with toothed stator poles and a permanent magnet rotor. Normally, they feature 200 rotor teeth and rotate at 1.8° per step. This is the best configuration, providing high torque and speed.

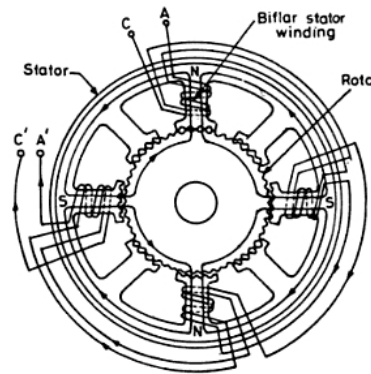


Figure 2.35: Hybrid stepper motor.

2.3.3 How to drive a Stepper Motor?

To drive a stepper motor, first thing to know is if it is unipolar or bipolar:

- **Unipolar** - it only has one winding per stator pole (Figure 2.36). Its movement can be reversed without inverting the current. Normally has four wires (both ends of the winding) but can have six (an extra wire per winding can be used for half stepping);
- **Bipolar** - features two identical sets of windings per pole (Figure 2.37). To obtain reverse movement the current must be inverted. Normally has six or eight wires.

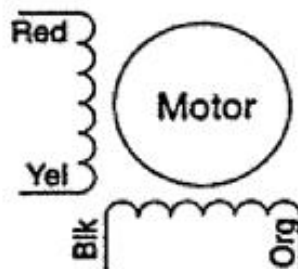


Figure 2.36: Unipolar stepper motor.

Depending on the type, different drivers can be designed. Figure 2.38 represents a typical configuration used to drive a stepper motor. Normally, a micro-controller is used to send the

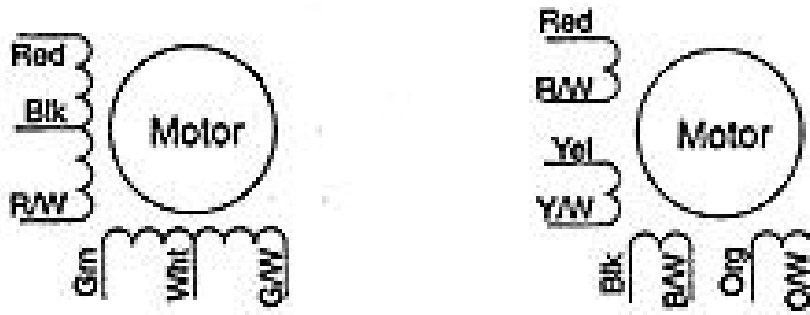


Figure 2.37: Bipolar stepper motor.

control signals to the power driver. The problem resumes to energize a winding and depending of the type, the driver might need to be able to reverse the current.



Figure 2.38: Stepper motor driver block diagram.

In Figure 2.39 we have an example of a $\frac{L}{R}$ drive for an unipolar motor. It is suitable for a low current stepper, of a few hundred mA , (if a stepper is to be used in the thesis work, it'll fall in this case) because some power will be lost in R_s (it is used to improve the rising time of the current). The micro-controller sends its signal to the transistor gate turning it on and off as needed [31]. One of this driver is needed for each winding.

The control should be performed closing each transistor in the order of the movement wanted, so one signal per phase is needed. More complex control can be used like half-stepping or micro-stepping for smaller steps.

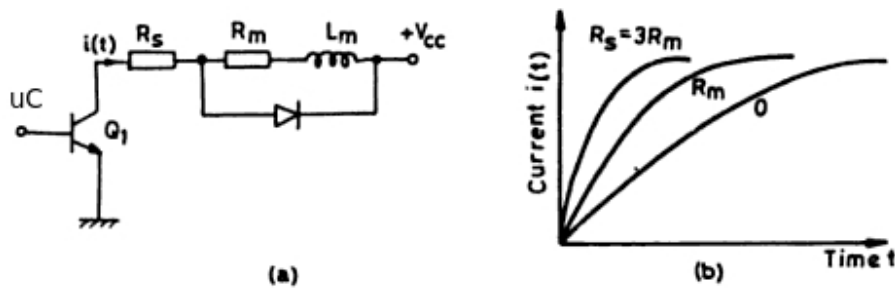


Figure 2.39: Unipolar stepper power driver circuit.

In Figure 2.40 we have an example of a H-bridge driver for an bipolar motor. It is suitable for any motor with any rated current because it can be adjusted easily. If SW_1 and SW_4 are closed and the other ones are open, the current is positive and the motor rotates clockwise (if coupled with correct actuation of the other windings). If the opposite is verified, the motor rotates counter-clockwise [31]

The control should be performed closing each pair of transistors in the order of the movement wanted. Contrary to the unipolar example, two signals are needed per phase due to the four transistors (they are controlled in pairs).

This driver is not worth doing with individual components, there are integrated circuits that already come with the H-bridge for each winding of the motor and their cost is really low (i.e. L6506D costs $\sim 6\text{€}$ in Farnell and features SMD package).

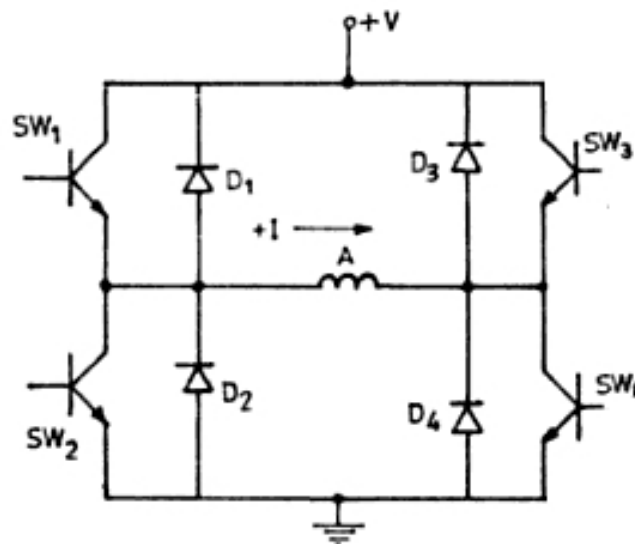


Figure 2.40: Bipolar stepper power driver circuit.

2.3.4 Remarks

This type of actuator is an interesting solution for the work in progress. It is very easy to drive (simple power drivers) and cheap. One big concern is the power consumption because the torque is proportional to it and that means an increased power supply if high torque is needed. To surpass this limitation, the load should be as low as possible.

Another issue is the step size that can be too high for this application. Higher resolution steppers cost more money and that can make this option not viable.

If this solution is to be used the best option is the hybrid motor because it has the best characteristics from the other options and features steps of 1.8° . Between unipolar or bipolar the option to take is to get the cheapest because the difference is only in the power driver. If an unipolar is used, the driver is really simple and can be made from individual components. If a bipolar is used,

a commercial integrated circuit is the solution and its price isn't so different from the other option that it makes no difference (already comes with the size to be incorporated in a PCB).

2.4 Piezoelectric Benders vs Stepper Motors

After all examples presented and analysed in previous points, is now time to select a route for the project. That's the subject of this section and to achieve this, a complete resume is presented and all pros and cons weighted.

Table 2.6 serves a starting point for this section. It presents all pros and cons of each technology.

Piezoelectric-Bending Step Motor		Hybrid Stepper Motor	
Pros	Cons	Pros	Cons
Cheap	Can't be applied directly (displacement is lower than necessary)	Cheap	Low resolution (compared to PZT solution)
High resolution	Complicated design (to surpass displacement limitations)	Simple design	Mechanical parts are needed
Low power consumption, works with any current (influences reaction time)	Low number of suppliers	High supply from big distributors	Power consumption (higher than PZT solution)
Novel application (One of the first motors with this limitations)	Novel application	High number of examples	

Table 2.6: Piezoelectric solution versus Stepper motor solution.

As shown in the table, both technologies are cheap. Despite that, the PZT benders are much harder to buy than the steppers. Any electronics distributor has a high selection of steppers motors contrary to the PZT actuators, only buzzers can be found. This is a big con for the PZT solution.

Comparing both solutions in terms of resolution and power consumption, the PZT solution clearly wins. It features resolutions in the order of μm while a stepper with that resolution would cost a lot more than the bender. The consumption of the PZT is really low, a current of $20\mu A$ can fully displace a bender [7] with $20\eta F$ in $1s$, while a stepper needs some mA to provide viable force.

Another con for the stepper solution is the need of complimentary mechanical parts to traduce the circular motion into linear. That will lead to increased cost and loss of reliability.

To complete the analysis, the number of examples found in literature must be addressed. Solutions based in PZT or Stepper technology can be found easily and in quantity but, while almost all stepper examples apply directly to the study, that doesn't happen with the PZT ones. Almost all are based on stack actuators and not a single one was found using the proposed solution. This happens because PZT solutions are mostly used when high precision is needed and cost is not a problem. So, that turns the proposed solution into a novel one, with a chance of creating and developing a new concept in linear motors, allying the precision of a PZT actuator with a focus in reducing the price in every way possible. But this brings a lack in documentation and examples too and that's why this is simultaneously a pro and a con.

Weighting everything, the choice falls into the PZT solution for now. Its pros clearly outweighs the cons even if it brings increased difficulty compared to the stepper solution. The challenge of trying to implement a new solution is really a force that must be taken into account and that will help to run over the bumps of the project.

2.5 Mechanical systems

This section aims to give some examples of xy systems that we can find in our everyday activity as well as some commercial options. Observing and studying these systems can provide valuable ideas and information to the project. These examples are presented below.

2.5.1 CD/DVD computer drive

A great example of a linear system is the CD/DVD-rom drive from any computer. It feature one axis and a motor (stepper or DC) that simply sends the drive out to allow the insertion of the CD.



Figure 2.41: CD drive examples.

These systems normally use a gear together with an toothed surface to translate the rotational movement to linear. Some feature a sensor to signalize the end of the path and stop the motor. The platform is normally hold by one side in a axis that is a iron/aluminium cylinder but some cases use two of these to make the system more robust. The point of contact with the axis is almost always lubricated for life use and made in a way that the friction is minimal. These characteristics can be observed in figure 2.41.

This systems are really simple and they can be used if a common motor is used.

2.5.2 Printers

Another common device that almost has at home is an Inkjet printer. The interesting part in these devices is the system that moves the the cartridges side to side. A motor (DC or stepper) is used to move them using a tensioned bets against a gear. See figure 2.42 and see the belt marked with number 258.

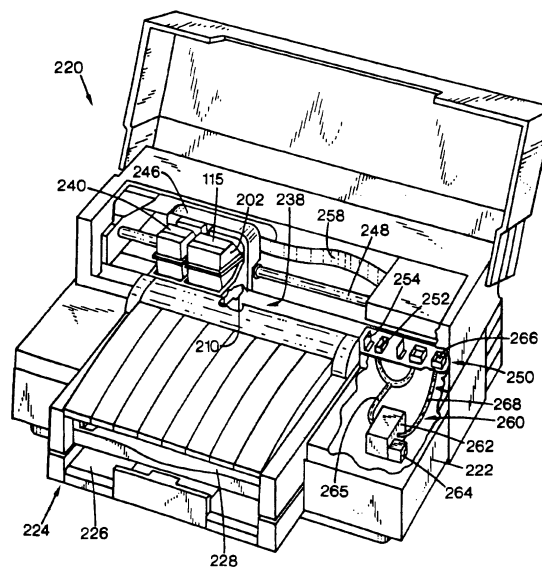


Figure 2.42: Printer example. Picture from HP Inkjet mechanism patent.

Actually, this is a system used in 3D printers too. It is really effective but needs several mechanical components (belts, sliding system, gears). That could prove to be a little too expensive if this type system is to be used.

2.5.3 Drylin Linear system

In a more industrial sense, Drylin linear systems are a product from Igus. Igus is a company specialized in polymers, bearings and linear systems. These systems are linear systems composed by a carriage lined by a proprietary polymer (iglidur J) with low friction coefficient and high

resistance to wear that requires no lubrication and a aluminium rail. They can be observed in figure 2.43.



Figure 2.43: Miniature Drylin T systems. Picture by Igus.

These systems would be ideal for the project because of their low friction but they are somewhat expensive. Assuming that the smallest system possible was used (the low-profile T line), the carriage would cost about 11.25 € and the rail 32.04 € per meter (prices by Igus in 26/06/2014). This goes against one of the main ideas of the thesis (doing it as cheap as possible) but stays as one available path.

These were only some systems that could be found and give great ideas for the project development. Certainly, there are a lot more examples and they will be explored if there is the need.

2.6 Smartphone - Arduino Communication

As said before in 1.3, a library to allow the communication between the Arduino and the smartphone needs to be developed. To allow the communication, the Arduino board must be able to connect to the smartphone as a normal USB device. For that, it must have built-in USB communication. The one that features this capacity is the Arduino Leonardo.

To prevent the creation from scratch, a search was conducted to find a backbone for it. It was made using Google and two promising results were found. Both results were open source, claimed that their solution permitted communication with the Arduino without any kind of rooting and that a full conversion between USB and serial was achieved. The next step was trying them and hope for positive results. The first solution, by Manuel di Cerbo [32], was provided in form of an example application (for both devices). After loading each file correctly, the solution didn't work. Some time was lost trying to find the problem but, after a few tries, the solution was rejected.

The second solution made by mike w (pseudonym, unable to find real name) was provided through an open repository in a form of a library [33]. Its name is *USB Serial for Android* and it has the basic functions implemented (`open()`, `read()`, `write()`, `close()`) and an example application is provided too. After loading both files, it worked. A communication was established and the data sent by the Arduino was printed on screen as shown in Figure 2.44.

```
Serial device:
CdcAcmSerialDriver
Read 9 bytes:
0x00000000 54 69 63 6B 20 23 37 34 0A
Tick.#74.
Read 9 bytes:
0x00000000 54 69 63 6B 20 23 37 35 0A
Tick.#75.
Read 9 bytes:
0x00000000 54 69 63 6B 20 23 37 36 0A
Tick.#76.
Read 2 bytes:
0x00000000 54 69
Ti
Read 7 bytes:
0x00000000 63 6B 20 23 37 37 0A
ck.#77.
Read 9 bytes:
0x00000000 54 69 63 6B 20 23 37 38 0A
Tick.#78.
Read 9 bytes:
0x00000000 54 69 63 6B 20 23 37 39 0A
Tick.#79.
```

Figure 2.44: Example application results.

The Arduino was programmed to print the following message in each second: "Tick.#XX" where XX represented a number incremented each second, starting in 0. The library for the project will be built over this one.

Chapter 3

Electronic Driver

This chapter is focused in the development of the electronic driver that constitutes the interface between the actuators and the smartphone application. Since two different actuators were used along the project development (see chapter 4), two different drivers are presented.

The presentation includes all the theoretic explanation, properly explained, starting with the component calculation and choosing, passing through the control/actuation and finishing with the PCB design. It includes all the experimental results (waveforms, actuator response) and final remarks too.

As a remainder, the smartphone used for the implementations was a Samsung Galaxy S4 Zoom and the controller board is an Arduino Leonardo.

3.1 Piezoelectric Bender Driver

3.1.1 Theoretical Study

3.1.1.1 Actuator Choice

As show in 2.2.5 and 2.2.6, the piezoelectric solution chosen was a bending actuator as a step motor (friction-based slip motor). This actuator features two possible topologies: serial or parallel (see 2.2.3.2). Since both options use almost the same driver, the choice was based on the price and availability. To address this, a market survey was conducted and several conclusions were taken:

- There are almost no retailers for this kind of PZT actuators. Several big companies were contacted (Mouser, Farnell, Digikey, etc.) and they only have PZT buzzers;
- Several producers were found namely American Piezo, Physik Instrumente, Noliac and Piezo Drive;
- At the date of the writing, only American Piezo and Piezo Drive had online shops with prices. All the others require that the interested person contact them;

- All companies were contacted. Using for reference a bender of $1mm$ of deflection, $\pm 100V$ and $2N$ of blocking force, the cheapest found was a serial bender (bimorph) by Piezo Drive priced at 13.52€;

The chosen actuator is a BA4510 [34] by Piezo Drive. It features a $2mm$ range of displacement at $\pm 100V$ and $2N$ of blocking force. Its capacity is $65nF$, physical dimensions of $45 \times 30 \times 10mm$, $1.61g$ and a resonance frequency of $170Hz$. This is a good solution because features high displacement at relatively low voltage and capacity. Since it is sold in an online store is easy to buy. The big con is that it is shipped from Australia making this a solution to be replaced in the future.

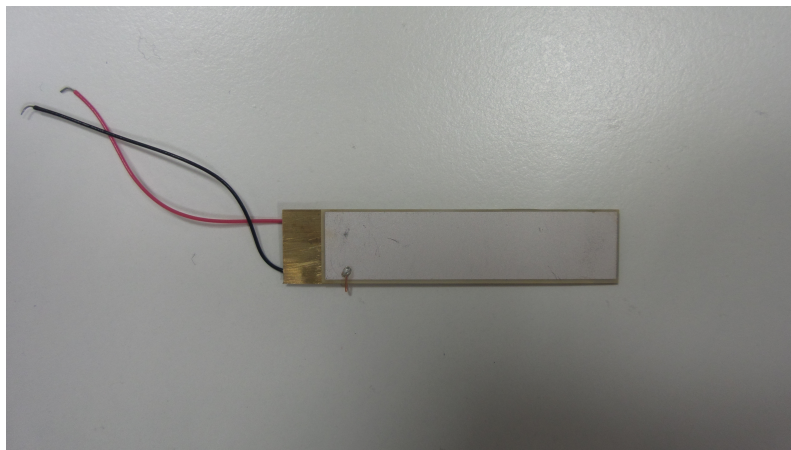


Figure 3.1: BA4510 PZT bender.

At least, two of this actuators are needed, one per axis.

3.1.1.2 Controller Choice

As said in 1.2, the controller that will be used must be Arduino based. This brings numerous advantages:

- Big online community. A big pro when problems arrive;
- Cheap and complete boards: features several ADC channels, lots of IO's, built in USB communication, etc;
- Very easy to program, with simplified C language and lots of libraries.

The only difficulty is to chose the version. The circuit will need at least 2 ADC channels (ideally 6) , 12 IO's and built-in USB communication (to be able to communicate directly with the smartphone). It must work with $5V$ too.

Analysing all the available Arduinos, the Leonardo proves to be the right choice. It has 6 ADC channels, 20 IO's, built-in USB communication, works at $5V$ and $16MHz$. Costs 18€+ VAT in the Arduino Store.

3.1.1.3 Boost Driver

Again, as shown in 2.2.5, the proposed solution for a bimorph bender is a two-stage boost converter followed by two h-bridges (one per actuator). This belongs to the DC/DC converters class and the circuit diagram without the h-bridges can be observed below.

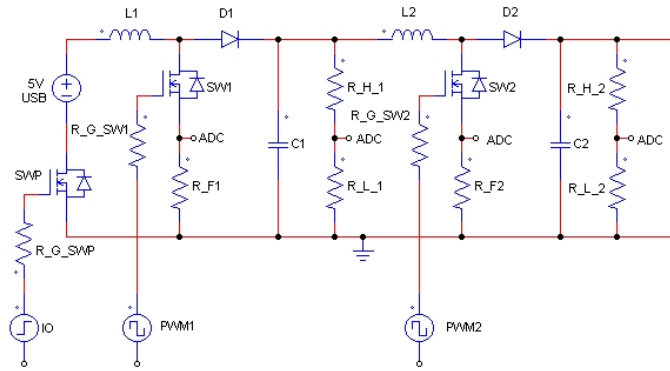


Figure 3.2: Proposed driver circuit diagram.

To analyse this circuit, we can just look to one stage because both work the same way, only with different components.

Looking for the first stage (ends in the resistors in front of $C1$), we can see that $SW1$ is the heart of this circuit and uses the principle of the power transfer between a coil and a capacitor that, contrary to a resistor, can store energy. The transistor is acting as a switch. Ignoring SWP , $RF1$, $RH1$, $RL1$, $RGSW1$ and $RGSWP$ for now (they aren't important for the converter's operation) and assuming that the $SW1$ and $SW2$ are working as switches (refer to figure 3.3):

- If the switch is closed, two circuits are formed: one composed by the 5V and the coil $L1$ and one composed by $C1$. The coil is charged by the input voltage and, because of the diode, the capacitor can only discharge for a load or, if no load is applied, theoretically, maintain its charge (1);
- If it is open, the full circuit is operating. The energy stored in the coil is used to charge the capacitor (2);

If the switch is operated fast enough, the capacitor doesn't discharge much and the coil never enters in the discontinuous mode (negative coil current) maintaining the desired voltage in the capacitor. The parameter used to control this circuit is the duty cycle of the switch that is nothing less than the time that it is open, in percentage, over a control cycle. This type of control is called pulse-width modulation (or PWM).

Before passing this information to valid equations and waveforms, some remarks should be made about this circuit:

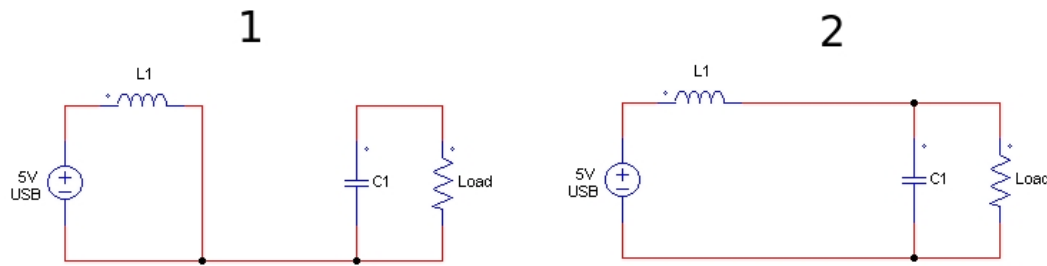


Figure 3.3: 1 - Coil charging, 2 - Capacitor charging.

- The power supplied to an eventual load is always less or equal to the power consumed in the input stage. Ideally is equal but since we have components along the way (diode, mosfet, coil, wires, etc.) that generate voltage drops, the output power will be always lower. Usually, the efficiency is about 80% [35];
- The current consumed by the driver is in spikes. Normally, this is bad for the power source;
- The control is really easy by a microcontroller. Only a PWM output is needed;

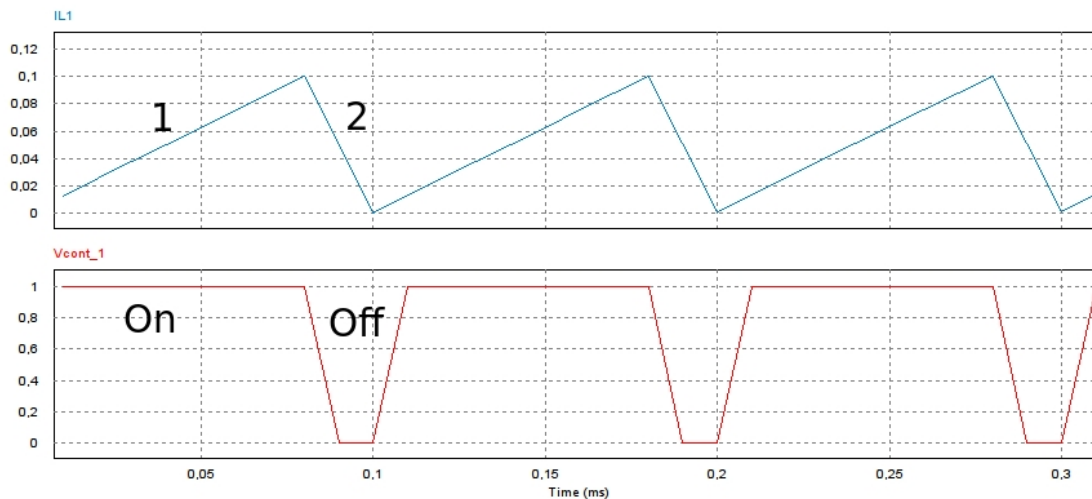


Figure 3.4: Coil current waveform

To deduce all the equations, the coil current waveform needs to be studied. In figure 3.4 we can observe the control of the switch and the reaction of the coil current. During the coil charge phase (when the switch is closed, numbered 1 in the wave) and assuming that $I_{l_{min}} = 0$ to prevent discontinuities, the current goes to a maximum value, $I_{l_{max}}$, and follows the equations:

$$Vl = L * \frac{\delta Il}{\delta t} \quad (3.1)$$

$$Il = \frac{1}{L} * \int_0^{d*T} Vl * dt \quad (3.2)$$

$$Il_{charge} = \frac{Vi}{L} * d * T \quad (3.3)$$

Where L is the inductance of the coil, d is the duty cycle and T is the period of control. As said before, the duty cycle is a value between 0% and 100%. During the coil discharge phase (when the switch is closed, numbered 2 in the wave), we have:

$$Il = \frac{1}{L} * \int_{d*T}^T Vl * dt \quad (3.4)$$

$$Il_{discharge} = \frac{Vi - Vo}{L} * (1 - d) * T \quad (3.5)$$

Assuming that the converter works in steady-state mode, the energy stored in the coil is the same at the start and end of the period. This only means that $Il_{charge} = Il_{discharge}$ and:

$$Il_{charge} = Il_{discharge} \quad (3.6)$$

$$\frac{Vi}{L} * d * T = \frac{Vi - Vo}{L} * (1 - d) * T \quad (3.7)$$

$$Vo = \frac{Vi}{1 - d} \quad (3.8)$$

$$d = \frac{Vo - Vi}{Vo} \quad (3.9)$$

This is the base formula of the converter and guarantees that the output voltage is always higher than the input voltage. Theoretically, this equation allows an infinite output voltage but the duty cycle will be excessively high. A maximum valid value for it is around 85% [35].

Using again the idea of the energy conservation, we can deduct the remaining equations needed to understand the functionality of the converter:

$$Il_{med} = \frac{Il_{max} - Il_{min}}{2} \gg Il_{min} = 0 \quad (3.10)$$

$$Il_{med} = \frac{Il_{max}}{2} \quad (3.11)$$

$$Pi = Po \gg Vi * Il_{med} = Vo * Io_{med} \quad (3.12)$$

Where Il_{med} is the medium coil current, Pi/Po the input/output power and Io_{med} the medium output current.

After the duty cycle and power equations reached, now the equations for the capacity of the capacitor and the inductance of the coil need to be deducted. For the coil, is really easy. Using the equations in 3.3 and evidencing L :

$$L = \frac{V_i * d * T}{I_{l_{max}}} \quad (3.13)$$

Choosing the maximum current consumed in the input, L can be adjusted as desired. For the capacitance, the capacitor current waveform must be studied. It can be observed in figure 3.5.

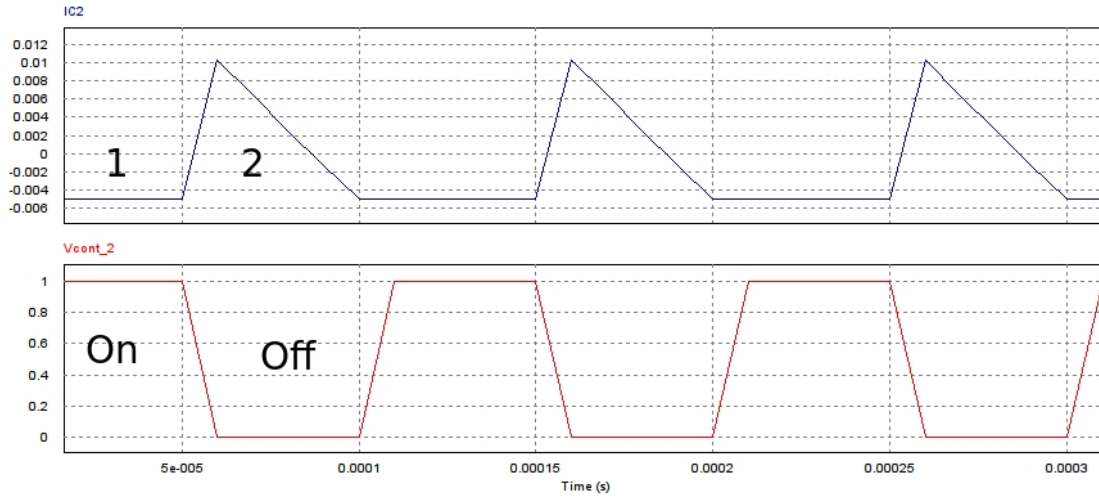


Figure 3.5: Capacitor current waveform

The area marked with 1 represents the discharge to the load. The area marked with 2 represents the charge provided by the coil. In steady-state, the two areas need to be equal. With the use of some knowledge in electromagnetism:

$$\int dq = \int i * dt \quad (3.14)$$

Where q represents charge. The integral represents an area and the equation for the capacitor current is $I_c = C * \frac{dV_c}{dt}$ so:

$$\Delta Q_c = C * \Delta V_c \quad (3.15)$$

$$C = \frac{d * T * I_{omed}}{\Delta V_c} \quad (3.16)$$

The ΔV_c represents the ripple in the output voltage. This can be chosen and adjusted as desired.

This concludes the formula deduction for the converter. For the second stage, everything is the same except that the input source is the previous stage. Nothing else changes.

As shown, several parameters must be defined before any component calculation. After that definition, everything can be calculated and that is presented next.

3.1.1.4 Boost Driver Basic Components

Several parameters need definition. They are the converter frequency, the output voltage (and its ripple) and the maximum current consumed:

- **Frequency** - It must be as high as possible to improve the driver response and efficiency. It is limited by the microcontroller's ADC speed (slowest part of the digital processing chain). The Arduino Leonardo works at 16MHz and the ADC channels can work at a maximum of 1MHz [36] but the higher the speed, lower the resolution. Choosing a value of 500KHz and knowing that each ADC read takes about 13 cycles (according to the datasheet), this represents ~ 38000 samples/second. If we set a frequency of 10KHz for the converter, this means that we can make at least 3 ADC reads per cycle. This will be important and will be discussed later in this document;
- **Output voltage and ripple** - since the PZT bender works at $\pm 100\text{V}$, that could be the output voltage. But since that, at maximum voltage, the force of the bender is almost 0N (see 2.2.2), the chosen output voltage is 50V that guarantees half of the blocking force. Using equation 3.9, this gives a duty cycle of 90%. This is too high and, if more voltage needed, the converter won't be able to provide it. This is the main reason for using two stages. The first stage will up the voltage to 25V ($d = 80\%$). The second will provide the 50V with a duty cycle of 50% that is acceptable and gives room for higher voltage values. An acceptable ripple is about 0.5V , up or down;
- **Maximum current consumed** - since there isn't a standard maximum current for USB-OTG, this driver needs to be as flexible as possible to provide the chance to change smart-phones if needed. Over 80% of the OTG-ready provide only a constant 100mA [37] and that is the value to be taken in account. Using 100mA and the equations 3.11 and 3.12 , $I_{l_{med}} = 50\text{mA}$, $P_{input} = 5\text{V} * 50\text{mA} = 0.25\text{W}$ and $I_{o_{med}} = 10\text{mA}$. As for the second stage, since the input voltage comes from a capacitor, it can have an $I_{l2_{max}} = I_{o1_{med}} * 2$ without too much voltage drop. So $I_{l2_{max}} = 20\text{mA}$;
- The Arduino consumption needs to be taken in account too but since the converter consumption is in spikes, the power bus just "sees" the medium current. This leaves 50mA for each component.

To sum everything up see the table 3.1.

Next step is to calculate the capacitors and inductances of the converter. Since all the formulas were already presented, this is just a mathematical exercise. The values are presented in table 3.2.

With all those components defined and calculated, there is enough information to perform some simulations to verify everything presented.

	$F(KHz)$	$V_o(V)$	$d(\%)$	$\Delta V_o(V)$	$I_{l_{max}}$	$I_{o_{med}}$	$P(W)$
Stage 1	10	25	80	0.5	100mA	10mA	0,25W
Stage 2		50	50	0.5	20mA	5mA	

Table 3.1: Piezoelectric driver ratings.

	$C(F)$ (3.16)	$L(H)$ (3.13)
Stage 1	4 μ	4m
Stage 2	1.25 μ	65m

Table 3.2: Piezoelectric driver capacitor and inductance values.

3.1.1.5 Boost Driver Simulations

For this preliminary simulation, a program named PSIM was used. This software performs a simulation different from the normal SPICE. It just focus in the more general aspects, ignoring, for example the transistor gate current needs.

With that in mind, the model used is presented in figure 3.6.

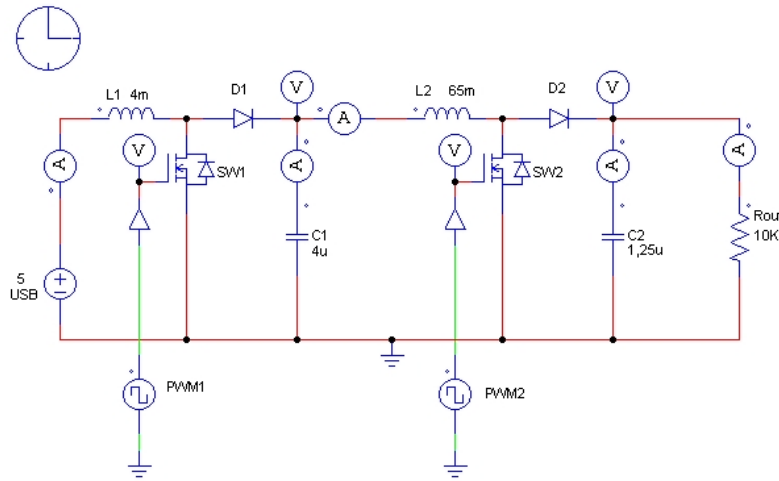


Figure 3.6: PSIM converter model.

Some configurations were made to guarantee that the converter was working in steady-state:

- Set the capacitor initial voltages to 25V and 50V respectively;
- Force a constant output current of 5mA with a 10K Ω resistor;
- Set a square wave with the required duty cycles and frequency in the gate of the transistor.

With this model and configurations, waveforms can be observed in the next figures.

Looking at the figures, some conclusions can be taken:

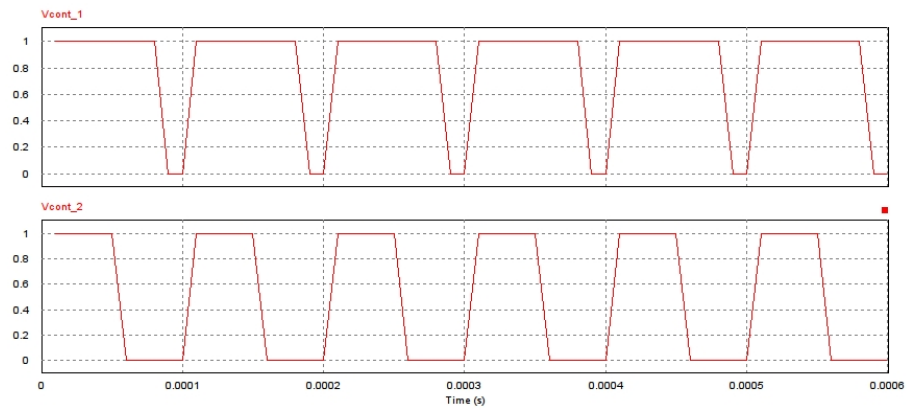


Figure 3.7: PWM waves. V_{cont_1} is the PWM for the first stage and V_{cont_2} is the PWM for the second one.

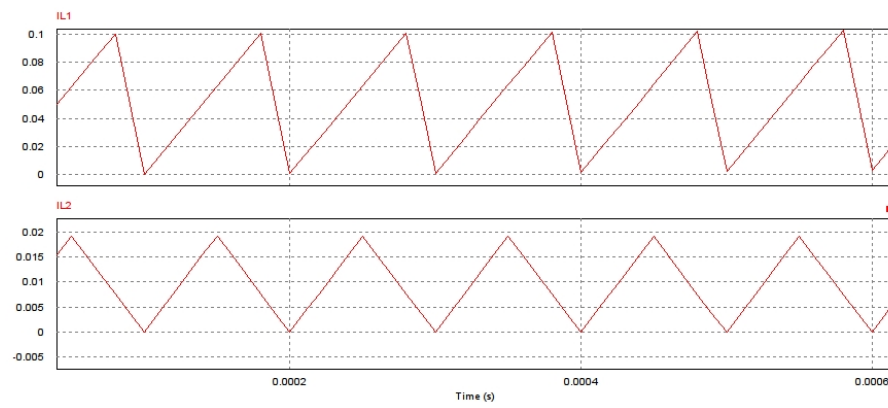


Figure 3.8: Coil currents. I_{L1} is the current from the first coil and I_{L2} is the current from the second one.

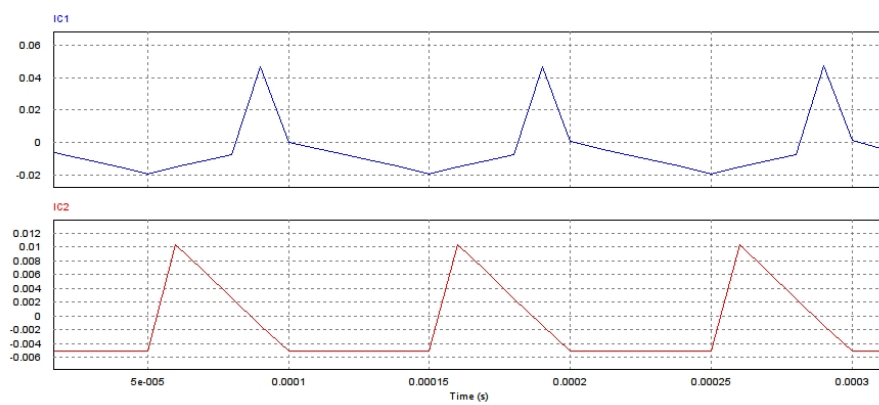


Figure 3.9: Capacitor currents. I_{C1} is the current from the first capacitor and I_{C2} is the current from the second one.

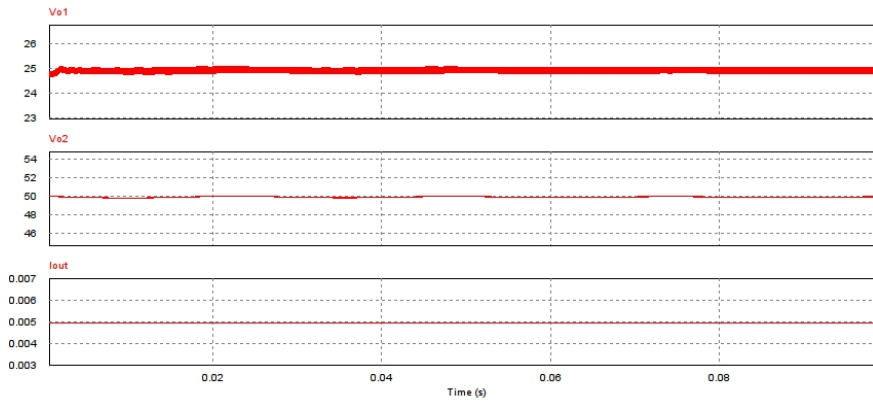


Figure 3.10: Output waveforms. Vo_1 is the output voltage of the first stage, Vo_2 is the output voltage of the second one and I_{out} is the output current.

- In figure 3.7 are presented the waves of control. The duty cycle is represented by the time that this wave is in the value 1 over a cycle. Comparing both waves, one can see that V_{cont2} has a lower duty cycle and around 50% as desired;
- In figure 3.8, can be observed that $I_{l1_{max}}$ is 100mA, $I_{l2_{max}}$ is almost 20mA and both never go below 0 as expected;
- In figure 3.9, the phases of charging and discharging can be observed clearly. When the current is negative, the capacitor is discharging to the load and in I_{c1} , the current wave is spiked because the second stage consumes it in spikes too whereas I_{c2} is constant (resistor). When the current is positive, the behaviour is almost the same in the two stages and represents the trade of energy from the coil to the capacitor;
- Finally, in figure 3.10, the output waveforms can be observed. Vo_1 presents some ripple as expected but stays around 25V while Vo_2 performs the same and with less ripple. I_{out} is constant.

With that simulation, the converter was validated and is ready to be implemented. The simplicity and flexibility was the main idea in choosing this topology. Since the control of the PZT bender is not well defined (it depends of the mechanical implementation of the stage), this converter allows the output of any tension from 5V up to 100V or even higher, allowing an adaptable control of the actuator. It allows re-usability for other applications too (stepper motors, changing the coils for example), making this a safe bet. Nevertheless, this flexibility comes with a price because it might not be the best choice for the stage depending of the actuation of the bender. This was a fair trade-off and was with that in mind that it was implemented, as presented next.

3.1.2 Implementation

For this step, the complete circuit must be studied because a full characterization of each component is needed. Looking to figure 3.2, there are several resistors that were ignored in the previous section. This was derived of their low impact in the driver operation:

- RH_x is used with RL_x to produce a voltage divider. This voltage divider is used to transform the output voltages of the capacitors in voltages that can be read by the microcontroller's ADC that works in 0/5V range. The resistors should be as high as possible to limit the losses;
- RF_x are resistors used as current sense to improve control, if needed. They will be of 1Ω and can be replaced by a wire;
- SW_x are the transistors used as switches. The available choices are between BJT, MOSFET or IGBT and will be discussed below;
- SWP is a simple on/off switch;
- RG_{xxx} are the gate resistors of the transistors. They must be calculated according to the transistor used and provide the needed current to switch it;
- Since the Dx are diodes, only the current and voltage ratings are needed to choose them.

Now that the other components functions are explained, the missing values need to be calculated.

3.1.2.1 Voltage Dividers

The equation for a simple voltage divider is presented in 3.17 and in figure 3.11.

The output voltage of the first stage is 25V. Setting this value to 4V for ADC measuring, this gives a ratio of 6.25. If $R2$ is setted to $1M\Omega$, $R1$ becomes $5.25M\Omega$.

The output voltage of the second stage is 50V but it can go higher. Setting 100V as maximum voltage value and making it 4V for ADC measuring, this gives a ratio of 25. If $R2$ is setted to $1M\Omega$, $R1$ becomes $24M\Omega$.

$$Vo = \frac{R2}{R2 + R1} * Vi \quad (3.17)$$

3.1.2.2 Transistor

With the ample choice offered in the market, it is really important to chose the right transistors for the application. There are three main hypothesis:

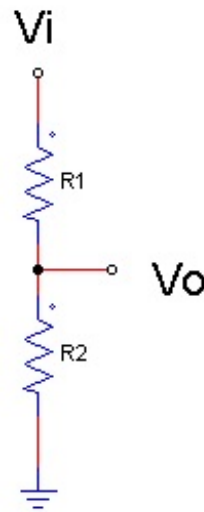


Figure 3.11: A simple voltage divider.

- A BJT requires a high base current to turn on, has a relatively slow turn-off but is really cheap because the technology is older and more matured;
- A MOSFET is a voltage controlled device that after a switch doesn't consume current. It can be operated at really fast frequencies, has a lower on resistance and has a built-in body-drain diode;
- A IGBT has the same conduction characteristics of a BJT but is voltage controlled like the MOSFET. However, the IGBT still has the disadvantage of a relatively slow turn-off.

Since the main objective is the efficiency, the solution is to use a transistor that can reduce the losses of the converter even if it has an higher cost. Using figure 3.12 information, the MOSFET is the best choice.

As for the gate resistors, they must be chosen just to limit the current that passes during the switches. Since the current will be supplied by the microcontroller, this is the limitation. According to the Arduino Leonardo datasheet, the maximum current per I/O is 40 mA. So a resistor of 220Ω is enough to limit the current (about $23mA$).

3.1.2.3 Component Ratings

With all main ratings presented and calculated, only the power ratings are needed before choosing all the components. Some of the ratings are easy to figure by just analysing the circuit but others need some calculations (medium or effective values). To prevent all those calculations, the values will be taken from PSIM simulations that already calculates them all. Table 3.3 resumes all of them in an easy to read form. All the values are for the worse cases ($V_{o1} = 25$, $V_{o2} = 50$ for the current and $V_{o2} = 100$ for the voltage).

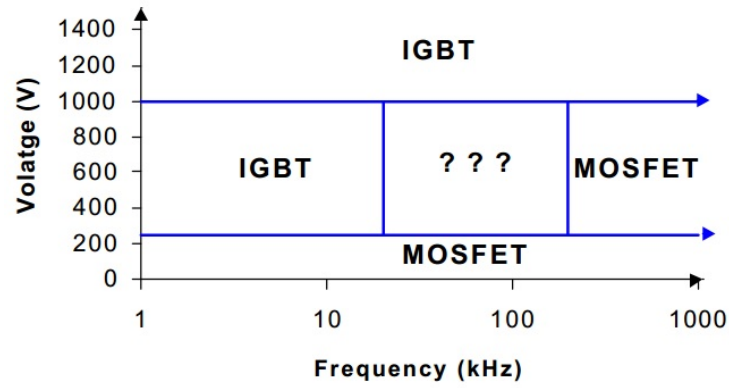


Figure 3.12: Semiconductor preferences according International Rectifier [9].

Component	Value	$V_{max}(V)$	$I_{max}(A)$	$I_{med}(A)$	$I_{rms}(A)$	Comments
L_1	4mH	25	100m	50m	67m	I_{medium} and I_{rms} are PSIM values
L_2	65mH	100	20m	10m	11m	
D_1	-	25	100m	7m	19m	
D_2	-	100	Depends of the load (minimum is 20m)	4m	6.8m	
SW_1	-	25	100m	52m	64m	
SW_2	-	100	20m	5.9m	9.2m	
SWP	-	5	100m	-	-	I_{medium} and I_{rms} don't really matter
C_1	4 μ	25	20m	-	-	
C_2	1.25 μ	100	Depends of the load (minimum is 20m)	-	-	
RG_{SWx}	220 Ω	5	23mA	-	-	
RH_1	5.25M Ω	25	4 μ	-	-	
RL_1	1M Ω	25	4 μ	-	-	
RH_2	24M Ω	100	4 μ	-	-	
RL_1	1M Ω	100	4 μ	-	-	

Table 3.3: Component ratings.

C_2 and D_2 maximum current depends of the load. Since it will be a piezoelectric component that is, basically a capacitor, the output current will be spiked and its maximum value dependent of the rest of the charge.

3.1.2.4 H-bridges

The H-bridges were already present in subsection 2.2.5. As for its implementation, there were two solutions in discussion:

- Make two custom H-bridges with each individual component (MOSFET, gate resistors and diodes);
- Use an IC that already includes all components needed;

Both solutions have their pros and cons. The first hypothesis presents better flexibility because it can be tailor made for the converter but at the cost of space and increased complexity. It would need four N-Channel MOSFETs, two P-Channel MOSFETs and one gate resistor per MOSFET. Multiply everything by two because of the second actuator. It would present the configuration observed in figure 3.13. The extra N-MOSFETs are used to drive the high side MOSFETs because of the floating voltage value in the middle of the bridge. This configuration has significant energy losses because of the pull-up resistors in the high side. An IC for the high side could be used to simplify this design. Looking in Mouser catalogue, we find the BTS4140N that supports voltage up to 60V, 900mA, on-resistance of about 1Ω but high power consumption at about 1.05 €. Looking for the other MOSFETs, the IRLML2060TRPBF appears. It supports 60V, 1.2A, has a on-resistance of $480m\Omega$ and costs 0.288 €, add more 0.1 € for the gate resistance. This totals 2.876 €. For two bridges, 5.752€.

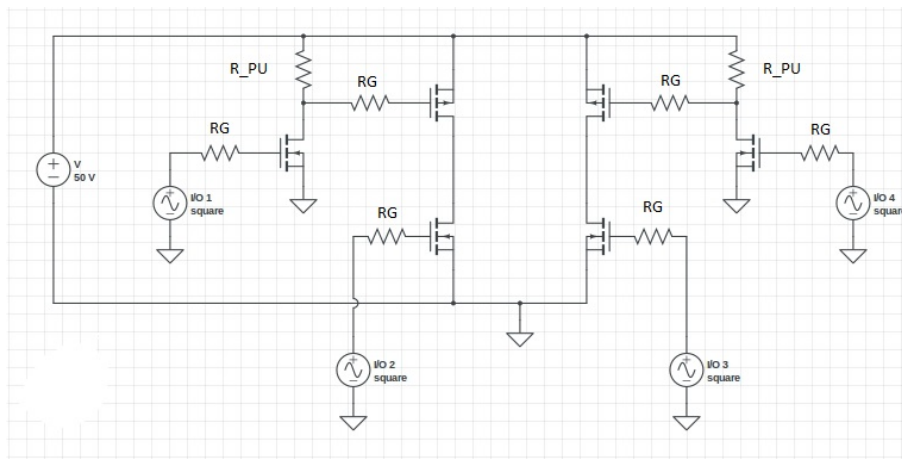


Figure 3.13: Mixed channels MOSFET h-bridge.

The second hypothesis, uses a commercial IC like the DRV8844. This IC includes all the necessary components to the correct functioning of the h-bridges, supports voltage up to 60V, 2.5A of continuous current, an on-resistance of about 0.5Ω and features several interesting functions like sleep mode, fault detection and reset pin. One of the disadvantages of this chip is that the lowest voltage supported by the bridge is 8V. So, the range is from $-50V$ to $-8V$, then $0V$ and from $8V$ to $50V$. This means that the lowest displacement possible is $80\mu m$. This doesn't seem to be a great handicap, although.

At the time of the writing, this chip was priced at 5.04€ in Mouser website. It costs less than the first hypothesis with a better performance (easy drive, very low on-resistance, low power

consumption) and low space occupation. The pros offered outweigh the flexibility of the other solution and the DRV8844 was chosen.

3.1.2.5 Components Ordered

After everything characterized, the components were defined. The supplier chosen was Mouser Electronics because they have very good prices and fast delivery. Table 3.4 include every component ordered.

They were chosen with the ratings presented previously and with focus in the PCB implementation regarding the packages. The package 1206 was chosen to be the minimum for the resistors because, with a package lower than that, the solder process becomes a burden. The other components' packages were chosen to be as common as possible to allow posterior changes.

Component	Reference	Rating	Package	Price (€)
L_1	SRF0905-202Y	$2 * 2mH$	Custom	0.864
L_2	B82731T2351A020	$68mH$	Custom	2.32
C_1	VZH-100M1HTR-0606	$10\mu F$	Custom	0.216
C_2	VEJ-220M2ETR-1313	$22\mu F$	Custom	0.744
D_1	MMBD914-V-GS08	$200mA$	SOT-23	0.072
D_2	GSD2004W-V-GS08	$225mA$	SOD-123	0.08
SW_1	IRLML2060TRPBF	$60V, 1.2A,$	SOT-23	0.288
SW_2	SI1480DH-T1-GE3	$100V, 2.6A$	SOT-363	0.244
SWP	SI2302CDS-T1-E3	$20V, 2.6A$	SOT-23	0.12
OpAmp	LMV324IDR	$5V, Rail -$ $to -$ $Rail, 4ch$	SO-14	0.64
OpAmp	LMV358IDR	$5V, Rail -$ $to -$ $Rail, 2ch$	SO-08	0.64
H-bridges	DRV8844PWP	$60V, 2.5A$	HTSSOP-28	5.4
C_{pump}	VJ1206Y103KXBPW1BC	$0.01\mu F, 100V$	1206	0.056
C_{vcc}	GRM31CR61H106KA12L	$10\mu F, 50V$	1206	0.52
C_{vcp}	VJ1206Y104KXJPW1BC	$0.1\mu F, 16V$	1206	0.048
14 Resistors	Multiple	Multiple	1206	0.896 (0.064 each)
2 Resistors	SR732BTDD1R00F	1Ω	1206	0.48 (0.24each)
Headers	Multiple	-	1 inch	3.288
Micro B USB	10118192-0001LF	$1.8A$	-	0.28
Total				17.196

Table 3.4: Ordered components in Mouser Electronics.

3.1.2.6 PCB

Since the Arduino Leonardo board has a very defined form, the best way to design the PCB is to make it an Arduino shield. According to their website:

"Shields are boards that can be plugged on top of the Arduino PCB extending its capabilities. The different shields follow the same philosophy as the original toolkit: they are easy to mount, and cheap to produce" [38].

This idea fits very well in the project and makes the driver very robust. The schematics and the PCB design can be observed in figures 3.14 and in 3.15.

Regarding to everything explained until now, there are some differences in the circuit. The most visible one is the inclusion of the voltage buffers. Those are needed for the ADC reads because the voltage dividers have too much impedance and that makes impossible to read the voltage directly (too much settling time). Then two non-inverting Amp-Ops (see [35]) were inserted in the feedback resistors because the voltage would be too low to read (maximum voltage in R_{Feed1} is $200mV$ and maximum voltage in R_{Feed2} is $20mV$), giving R_{Feed1} voltage a gain of 13.33 (maximum current corresponds to $13.33 * 0.1 = 1.33V$) and R_{Feed2} voltage a gain of 100 (maximum current corresponds to $100 * 0.02 = 2V$).

Some pull-down resistors were inserted too in the outputs of the microcontroller for increased safety. The rest was already discussed.

The PCB was divided in several parts. There is a $5V$ and ADC part where all Amp-Ops are as near as possible from the ADC pins to reduce the noise from the rest of the circuit. There is a $25V$ section where everything related to the first stage of the converter is located. Finally, there is the section of the $50V$ where everything that works at that voltage is located.

The input voltage comes from a micro-usb connector where only the $+V$ and GND are connected. There is a dedicated (V_{test}) connector to test the capacitor voltages and to permit the driver expansion. Finally, there is a dedicated connector to the motors.

Both top and bottom layers were used, making this a two layer design.

The PCB was validated and manufactured in PCB Pool. Since their lowest work area is $1dm^2$ and the PCB has $65.58 \times 53.5mm$, the order was made for two boards. This came to a total of $116.33€$. The PCB, in stripped and ready forms, is presented in the figure 3.16 and 3.17.

3.1.2.7 Arduino Control

To finish the implementation, the Arduino firmware is needed. In this section, the communications will not be discussed, they are discussed in section 5.5.

The Arduino needs to control the three parts of the converter (first DC/DC stage, second stage and the H-bridges):

- **First DC/DC stage** - there are four tasks in this stage:
 - Control SWP - this switch is like an On/Off switch, it is turned on when the converter needs to operate and off when the converter isn't needed;

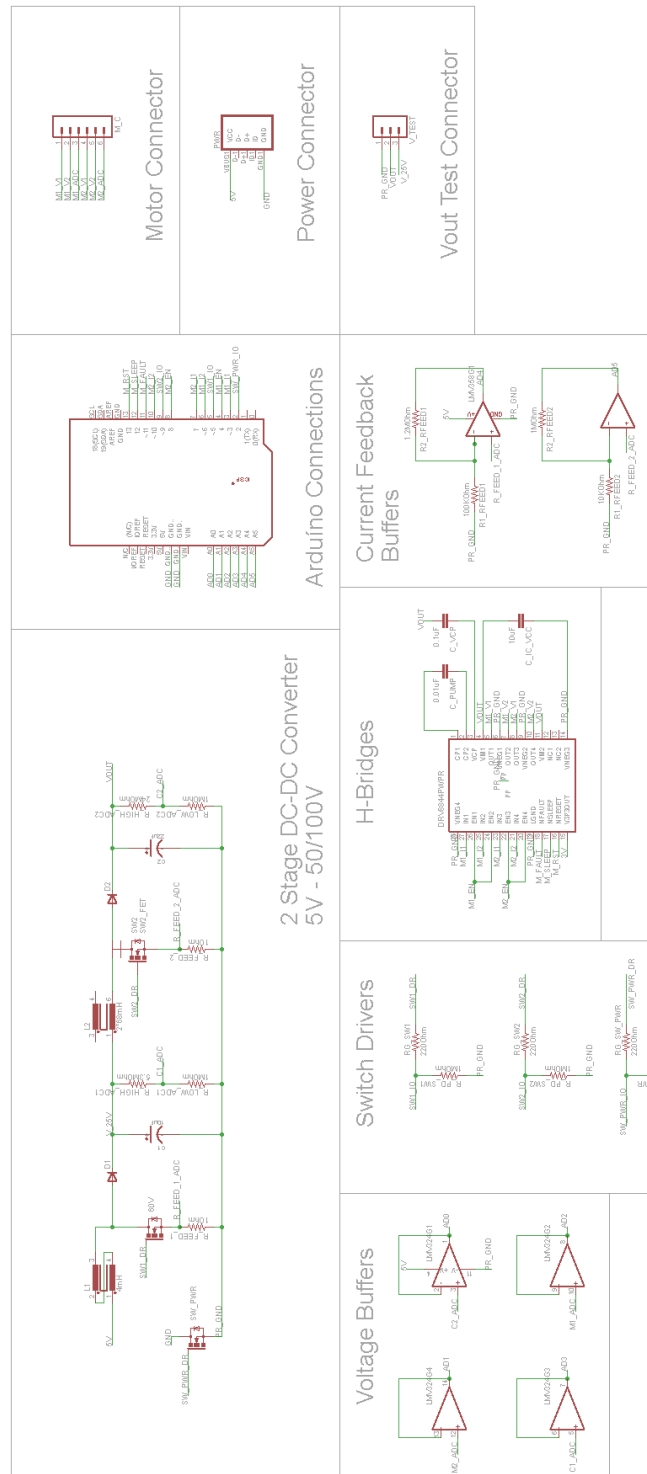


Figure 3.14: Driver schematics.

- Control SW_1 - here will inserted the first PWM signal. Using one of the PWM pins, the switch operates with the desired duty cycle. The duty cycle will always be 80% when more than 25V are needed and lower if less that 25V is needed (always following

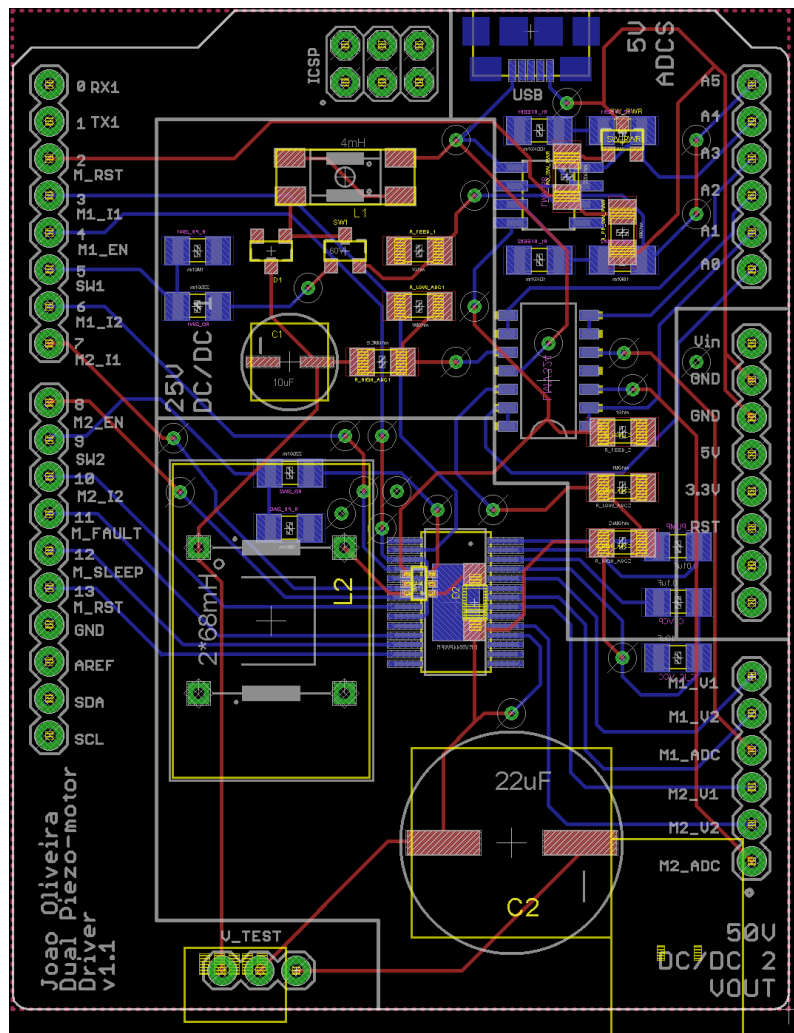


Figure 3.15: PCB layers.

the equation in 3.9). The duty cycle is setted to 0 whenever the output voltage is higher than what is expected (gives time to the capacitor to discharge a bit). This type of control is a bit rudimentary and should be changed in future work (to a PID for example). This option was taken to provide a working controller as soon as possible and allow the actuator tests;

- Read the current, $RFeed_1$ - at the time of the writing, this measure was ignored. Future work;
- Read the output voltage, ADC_1 - the values read are used by the controller to set the duty cycle of the switch as explained in the previous item.
- **Second DC/DC stage** - this stage is almost equal to the previous one. The only thing that changes is the threshold for the duty cycle. If more than 25V are needed the duty cycle is adjusted to the correspondent value. If not, the duty cycle is setted to 0%;

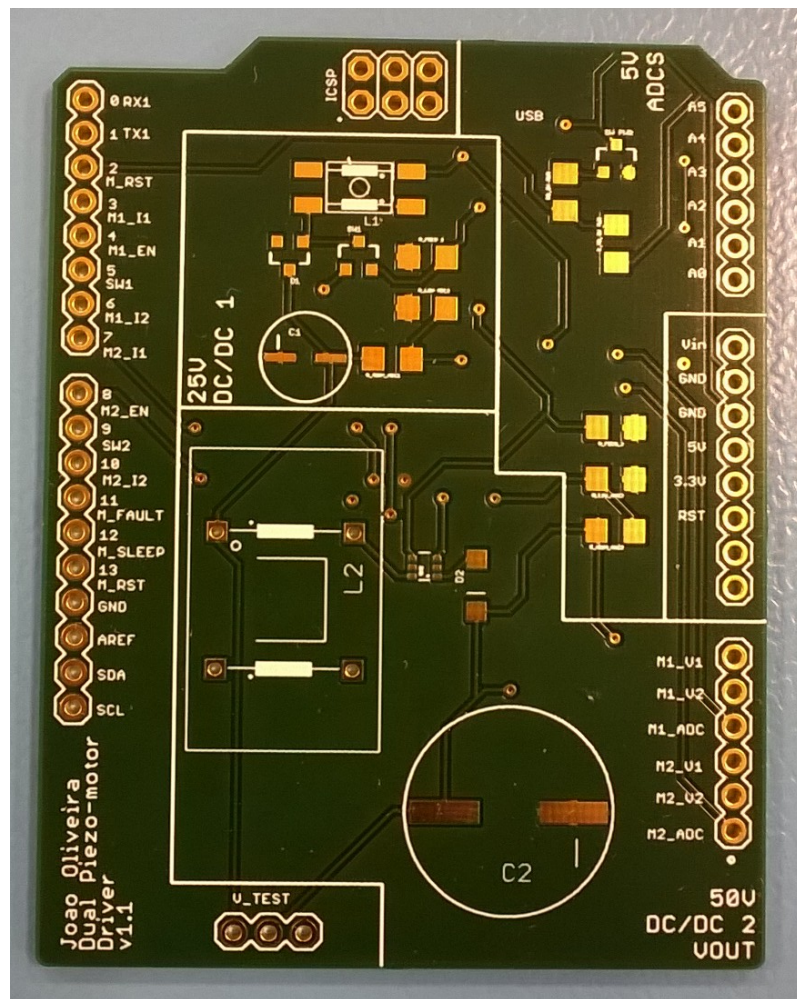


Figure 3.16: Stripper Boost Converter PCB.

- **H-bridge** - for the h-bridge, several pins need to be controlled. For each bridge, there are four essential pins, both EN_x and both IN_x . Those pins control each half bridge and the control logic can be observed in table 3.5. According to the position commanded by the smartphone, the inputs change as desired ($In_1 = HIGH, En_1 = HIGH, In_2 = LOW, En_2 = HIGH$ for $+Vo$, $In_1 = LOW, En_1 = HIGH, In_2 = HIGH, En_2 = HIGH$ for $-Vo$ and $En_1 = LOW, En_2 = LOW$ for $0V$). There are two more pins that need active operation, $nRESET$ and $nSLEEP$. The first resets the internal logic with LOW input and must be setted to $HIGH$ to allow normal operation. The second puts the device to sleep with LOW input and must be setted to $HIGH$ too. Finally, there is a pin that goes to LOW if there is any problem with the device, $nFAULT$. This bridge can be voltage or PWM controlled.

The controller firmware presented is very simple but performs rather well as will be presented next. This type of control was chosen because of its simple implementation and low development time (the biggest pro). Although, in the future, the controller should be changed to a more evolved and robust one (PID, for example). This could bring improvements in the output ripple and in

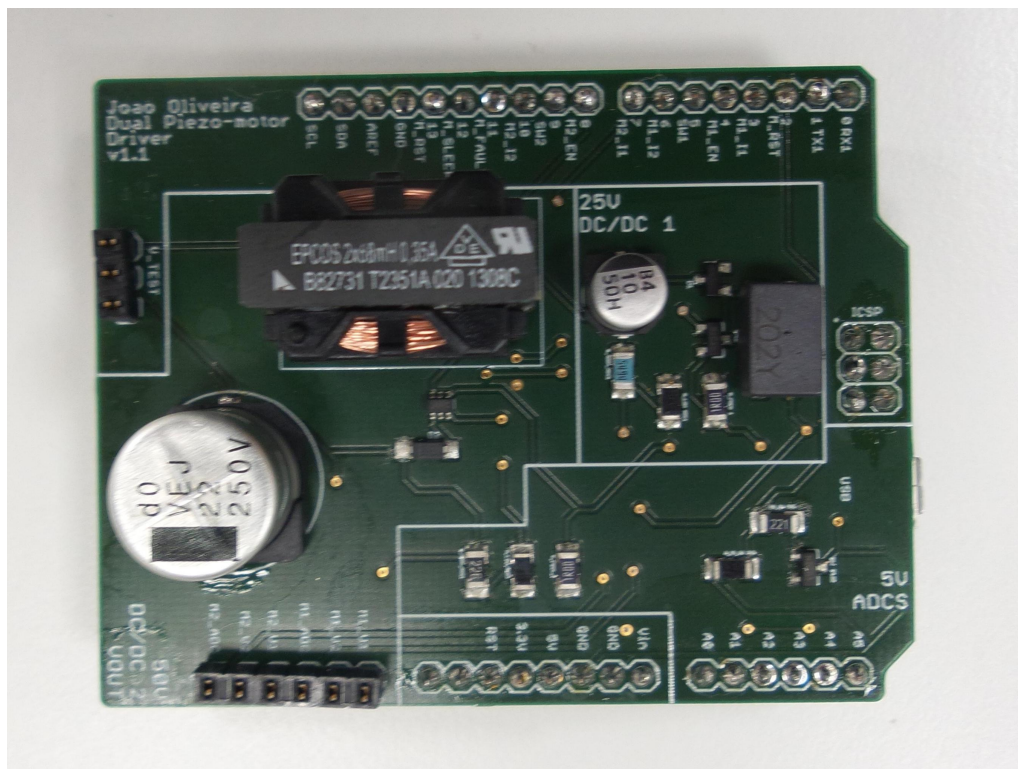


Figure 3.17: Boost Converter.

In_x	En_x	Out_x
x	0	Z
0	1	L
1	1	H

Table 3.5: H-bridge logic. Taken from the datasheet, property of Texas Instrument.

the power consumption but needs good adjustment and that takes time (and was not viable in this project time-frame).

3.1.3 Experimental Results

At the time of the writing, the PCB experimental results were not as expected. It wasn't able to elevate the voltage over 20V meaning that something was drawing too much current. The most probable causes could be:

- Bad programming of the firmware that lead to an incorrect functioning of the *DRV8844*;
- The *DRV8844* draws more current than what is advertised;
- There is an error in the PCB print/design;
- Wrong model of one of the components (the first coil, most probably);

- The Arduino is drawing some power from the driver (some current loop created);
- Some obscure PCB design rule that was not followed.

This was a serious drawback because the PCB was only ready two weeks before the deadline. This gave no time to perform the necessary debug of the board. Although, this was not a deal breaker because, while the PCB was not ready, the driver was implemented in a development board with some spare components that were lying in the laboratory. Although this was not ideal, it was working and drove the benders quite well. The circuit was not 100% faithful to the one presented but was really close. The differences were (compare with figure 3.14):

- No SW_{PWR} and R_{FEED1}/R_{FEED2} (meaning that no *OpAmp* too) were used;
- $C1$ is 63V 4.7 μF and $C2$ is 450V 2.2 μF ;
- The diodes are two 1N4003 (200V 1A);
- $L1$ is 4mH and $L2$ is 68mH;
- Every transistor was a BJT and a BD139 (NPN) or BD140 (PNP) with gate resistors of 11K Ω ;
- The H-bridge was made with individual components and follow the circuit show in figure 3.13 but with BJT transistors (the R_{PU} are 10M Ω resistors);
- The OpAmp used was a OPA2350;

To test the validity of the driver, a triangular wave was asked of the driver (see why in section 4) while using a bender and a 1K Ω resistor. The driver responded with the wave shown in figure 3.18.

The wave shows the triangles asked but the deadzone (the -5 to $+5V$ that is converted in 0V) can be well observed. This is one of the limitations of the driver but shouldn't be a big problem in the table implementations. It is even more noted because the bender doesn't discharge as fast as desired. This can be corrected with a bigger resistor charge but that will ask more current from the driver. It is a matter of calibration but will stay like this for now. With this test, the driver is validated.

3.1.3.1 Performance Analysis

As far as performance goes, two really important aspects can be tested:

- **Output Power** - the driver was tested with only a resistor as a load and without the h-bridge. Starting with an 200K Ω (250 μA) resistor, its value was successively lowered until the driver couldn't output the 50V. With that, a value of 20K Ω was achieved. This represents an output current of about 2.5mA, half of what was expected. This represents an efficiency of 50% and was somewhat expected because of the use of a development board. It features several

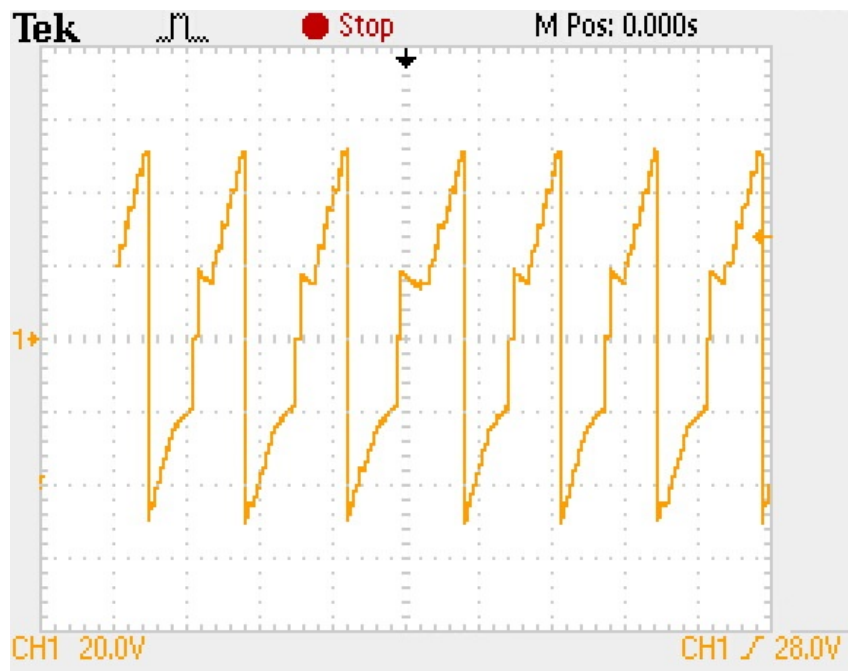


Figure 3.18: Wave response of the driver.

parasitic RLC and the components weren't the most indicated. This value can be seen as a success;

- **Ripple** - as said in the implementation section, the accepted value is 0.5V up or down. With the resistor specified above, a measure with an oscilloscope was made. It reported a maximum value of 50.4V and a minimum of 49.6V. It is between the gap defined. This measure can be observed in figure 3.19.

3.1.3.2 Consumption Test

Again, the used circuit was the one implemented in the development board, not the PCB one. The values here are estimated to be higher if the PCB was working flawlessly because all the components were picked with power consumption in mind (just the MOSFETs are a big difference because they are voltage controlled not current like the BJTs).

As for the test, a very simple one was made after all the process of assembly (see section 4.6). The smartphone was charged up to 100% and a simple infinite loop of a full grid (10x10 steps, see figure 4.35) route was coded. Each step was made with 50V output voltage and a step interval of 5s.

An Android application named *GSam Battery Monitor* was used to monitor the battery consumption. This app give numerous information but the most useful for this test is the battery percentage and voltage level. The app is shown in figure 3.20.



Figure 3.19: Output voltage of the driver.

For this exercise, we assume that the discharge of the battery is linear. This can be made because, during the discharge from maximum value to more or less 3V, the curve is almost linear as can be seen in figure 3.21.

The test was started at 22h39 and featured 4.269V and 100%. After 1h30 the test was stopped and the values were at 3.921V and 76%. Doing a bit of extrapolation:

$$m = \frac{4.269 - 3.921}{24\%} = 0.0145 \quad (3.18)$$

$$4.269 = m * 100 + b \Rightarrow b = 2.819 \quad (3.19)$$

This puts the lowest voltage at 2.819V. That's very unreasonable because the minimum value never should go below 3V [10]. The best way to correct this curve is using only the voltage levels. With that:

$$m = \frac{4.269 - 3}{100} = 0.01269 \quad (3.20)$$

$$4.269 = m * 100 + b \Rightarrow b = 3 \quad (3.21)$$

The plotted curves can be observed in figure 3.22. Using the voltage values and doing a simple

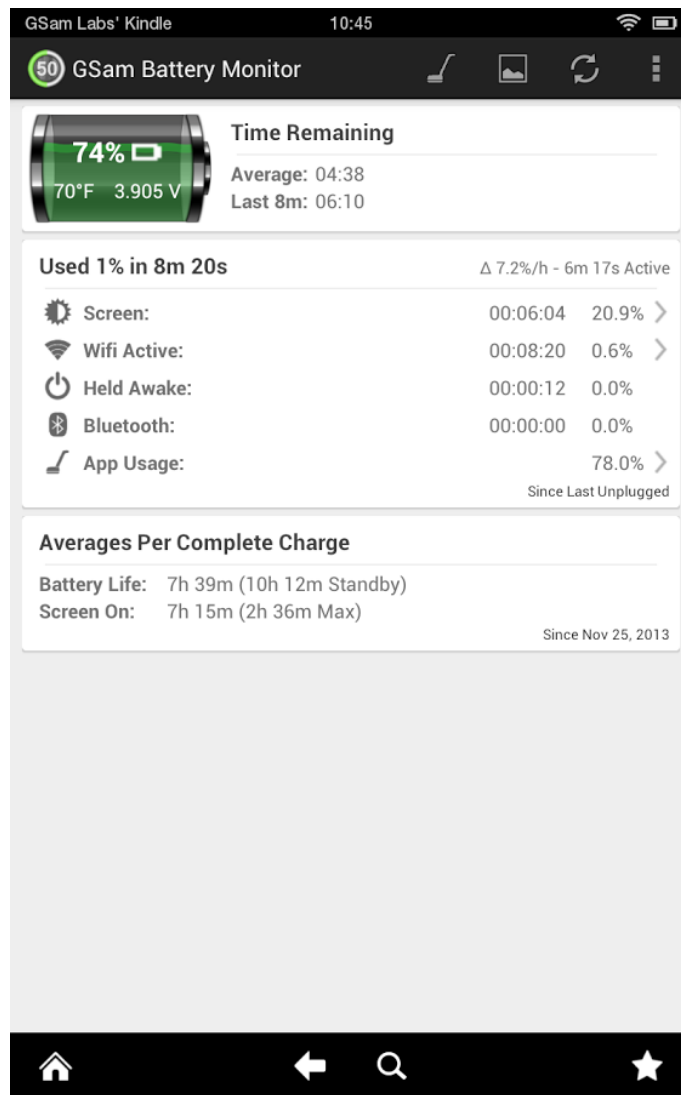


Figure 3.20: GSAM Battery Monitor.

proportion calculation:

$$4.269 - 3.921 = 0.348V \rightarrow 1.5h \Rightarrow \frac{(4.269 - 3) * 1.5}{0.348} = 5.47h = 5h 28min \quad (3.22)$$

The value in equation 3.22 represents the approximate autonomy of this system, assuming non-stop working. This is well above the 1h refereed in subsection 1.2.1 even if this is without the image processing tasks and lightning.

3.1.4 Remarks

After the driver presentation, some remarks can be made:

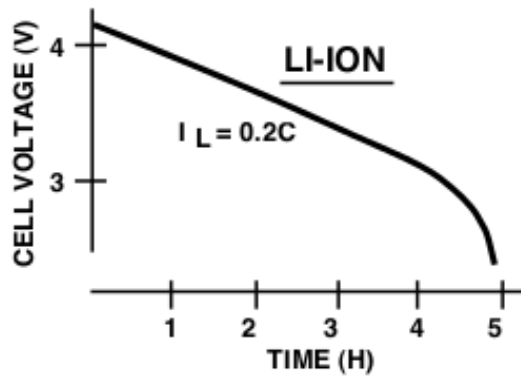


Figure 3.21: Li-Ion battery discharge plot [10].

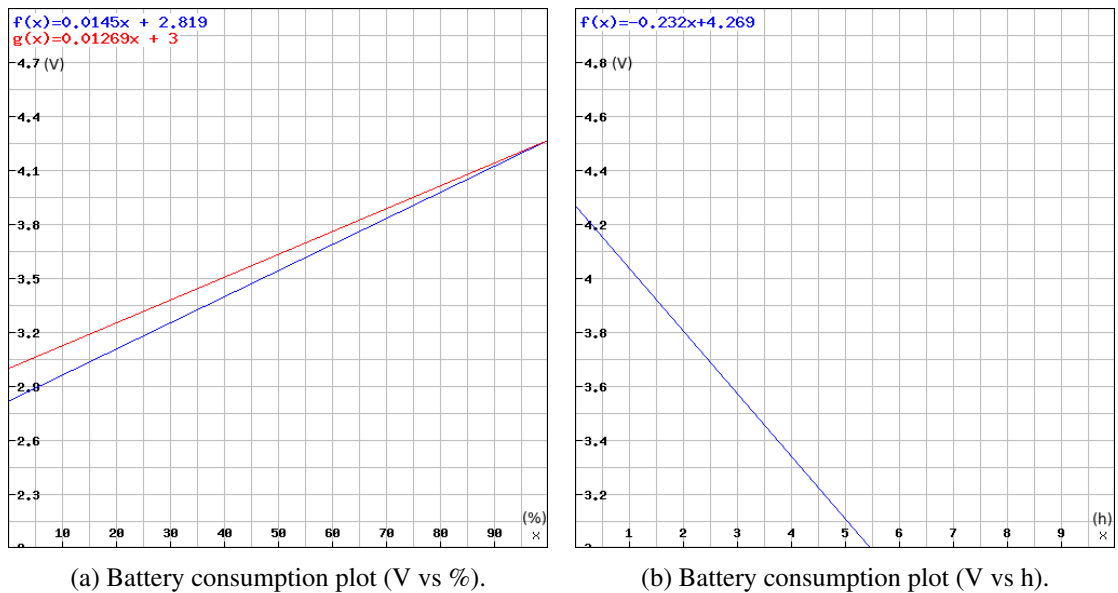


Figure 3.22: PZT bender driver consumption plots.

- The PCB didn't work. The reason wasn't found by the time of the writing but the expected reasons were enumerated and will be tested in future work;
- In other hand, the circuit worked as expected in a breadboard implementation. The numbers presented weren't as good as if implemented in the PCB but good enough to actuate the benders;
- The control used works but a better one should be developed in the future. Since it is only software, it doesn't condition the driver implementation. A PID controller is a good idea for it as shown in [39];

- The driver features some limitations. The deadzone is one of them and the bender discharge is the other. This brings the question if this is the best path for the implementation of a PZT driver... It is already known that this driver aimed to be as flexible as possible to allow some experimenting in the XY Table development but, when this implementation is well defined, a better, more focused driver should be developed;
- The power consumption was a bit high derived of the backup plan for the PCB. It featured *5h 28min* (extrapolated value) of autonomy and this value is estimated to rise with a successful PCB implementation;
- The driver construction costs 17.96€ for the components plus 116.33€ per two PCB boards. The full price is 76.125€. This is a prototype cost, in a large scale production, the cost can really go down. Using the PCB Pool price comparator, 100 PCBs cost 424.84€ and that makes each one cost 4.25 €, for example. Using the prices given by Mouser, if 100 drivers are made, the cost is 14.51 € per board. This totals 18.76 €.

3.2 DC Motor Driver

3.2.1 Theoretical Study

To drive a DC motor, only a h-bridge is needed per motor. The idea is very simple and explained in figure 3.23. If the 1 and 4 MOSFETs are closed, the motor spins to one side and the inverse happens if 2 and 3 are closed. Some resistors can be added if a fast current limiting method is needed.

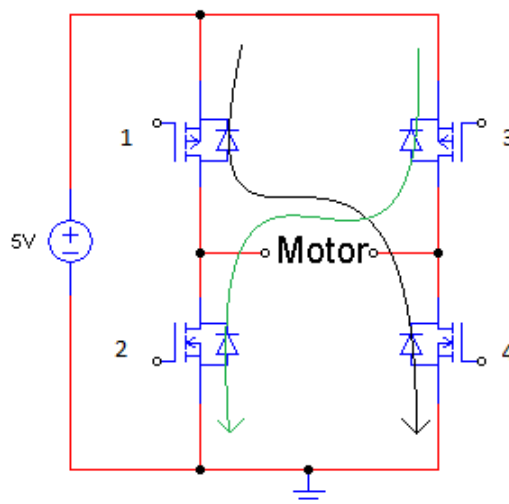


Figure 3.23: DC motor h-bridge.

This is a very simple circuit and allows a fast motor actuation with a very easy control. The maximum current allowed is the the OTG limit that depends on the smartphone used.

3.2.2 Implementation

Like the previous driver, there are two solutions for the application, one custom made and the other using a commercial IC. With a quick web research, we can find the *L293* family of circuits. These ICs are exactly what is needed because they feature two h-bridges, work at 5V with a maximum output current of 2A and are driven directly by the Arduino pins. They are very similar to the *DRV8844* presented before.

There are some variants of the same circuit, like the *L293D* that incorporates the fly-back diodes in the chip or the *L293E* that has current sense pins. It really comes to availability rather to functionality but, if all variants are available, the *L293D* is the best option because it saves space derived to the included fly-back diodes.

The used IC was the *L293E* (see table 3.6) and the circuit used can be observed in figure 3.24.

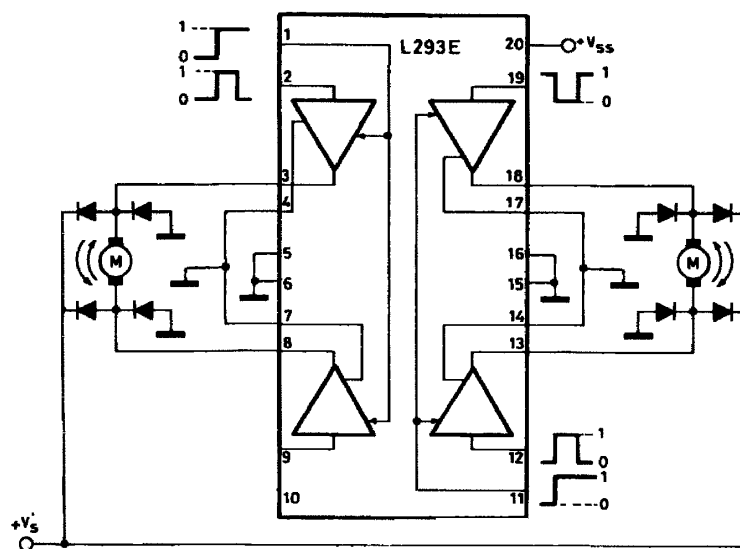


Figure 3.24: *L293E* h-bridge circuit.

Some $0.1\mu F$ ceramic capacitors were added in the motor pins to help stabilize the current. In V_{ss} pin a $0.1\mu F$ ceramic and a $100\mu F$ electrolytic capacitors were added to filter current spikes and voltage drops.

Since the Arduino Leonardo was already available, it was used for the implementation.

This implementation leaves the current control to the load. This allows the motor to be directly connected to the IC or with a current limiting resistor. If the resistor is used, it should be of at least 50Ω and $0.5W$, forcing the maximum current value to be $100mA$, respecting the value referred in subsection 3.1.1.4.

3.2.2.1 Components Ordered

Since this solution was developed without a lot of time remaining, the IC was bought in Aquário (Porto store) with a cost of 3 €. It was a *L293E* because the *L293D* was sold out so some diodes were needed too. There were some 1N4148 in the laboratory so those were used.

Some spare capacitors were used to stabilize the current as said above. There were spare headers in the laboratory so they were used too.

In table 3.6 there are all the components that are needed to make a PCB if ordered in Mouser.

Component	Reference	Rating	Package	Price (€)
$H_{bridges}$	L293DD	36V, 600mA	SO-20	2.94
4 Ceramic Capacitors	VJ1206Y104KXXCW1BC	0.1 μF	1206	0.192 (0.048 each)
2 Electrolityc Capacitors	VE-101M0JTR-0605	100 μF	SMD	0.24 (0.12 each)
Headers	Multiple	-	1 inch	3.288
Micro B USB	10118192-0001LF	1.8A	-	0.28
Total				6.94

Table 3.6: Components need for the DC motor driver if ordered in Mouser.

There were used some sensors that signalize the end of the path. Those were just two conductive pieces of metal that, when pressed on against each other, create a ground connection and that is read by the Arduino. Those sensors were be used to mark the point (0,0) of the grid.

3.2.2.2 PCB

There was no time to design a PCB for this driver. The solution was to use a solderable protoboard and create a cheap shield for the Arduino. It is a fast and easy way to turn the design more robust and to eliminate the need of a breadboard.

The circuit used can be observed in figure 3.25 and the protoboard developed in figure 3.26.

3.2.2.3 Arduino Control

The control is almost the same as the *DRV8844*. The big difference is that this bridge will always work at 5V and there are no special pins, only the In_x and En_x . The table 3.5 holds true for this IC as well.

The control will be made with PWM output and its duty cycle will be changeable by the smartphone. The frequency used is 10 KHz because it is under the interval reference for low power, hobbyist DC motors according to [40]. This frequency is more than enough to drive the motor since, at lower duty cycles (< 10%), it won't even turn. The PWM control allows smaller steps and that will be tested (see 4.5.2).

Since the circuit has two sensors to signalize the origin (see section 4.5), these are used to reset the position. In the control, if the backwards movement reaches the sensors, it is stopped.

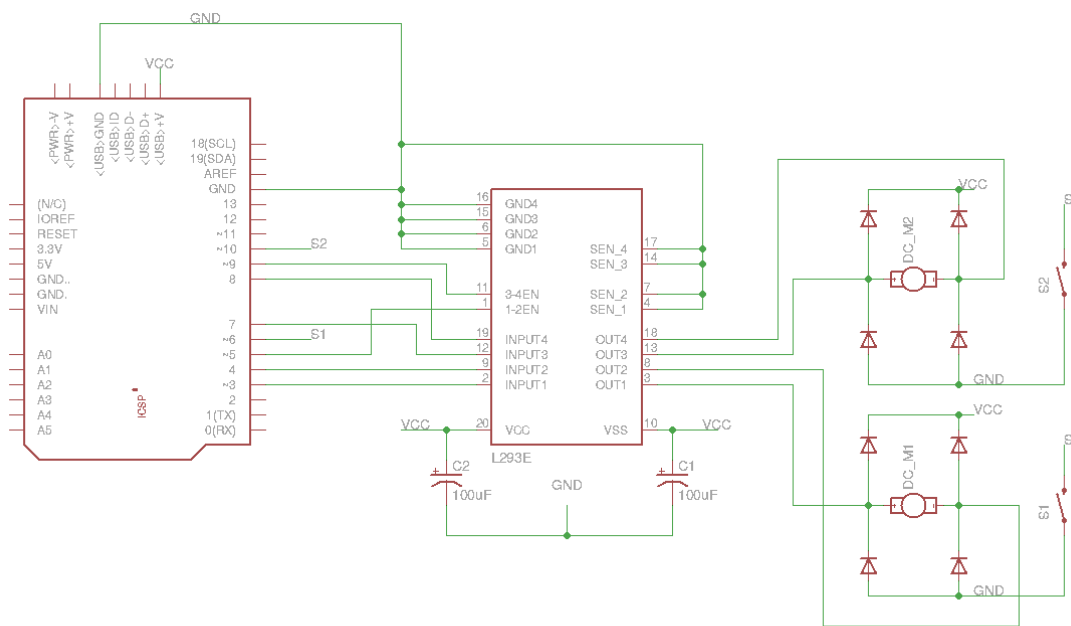


Figure 3.25: DC motor driver circuit.

The smartphone controls the time and direction of the movements. This is discussed in chapter 5. Everything resumes to the control of some outputs and two PWM pins.

Finally, only one motor should be driven at any given time because of the current limitation of the smartphone.

3.2.3 Experimental Results

After everything assembled, some tests were run to evaluate the validity of the circuit. No current limiting method was used because the motor maximum consumption was about $200mA$ (see section 4.5).

To test the validity, the smartphone commanded the Arduino to perform steps in one of the axis (without any load), spaced by 1s, with a duration of 20ms and 100% duty cycle. The circuit responded as expected and the waveform produced can be observed in figure 3.27. The waveform presented some noise. This was derived of the lack of filter capacitors in the motors line. They should be inserted in the future.

The waveform presents the times specified and has a good reaction time. This validates the circuit presented (and the smartphone application but more about that in chapter 5), allowing the time analysis and the consumption test presented next.

3.2.3.1 Time Analysis

This analysis aims to discuss and present the time rating of the system, from the order to the real actuation. For that, some time values need to be referred (the communication values are

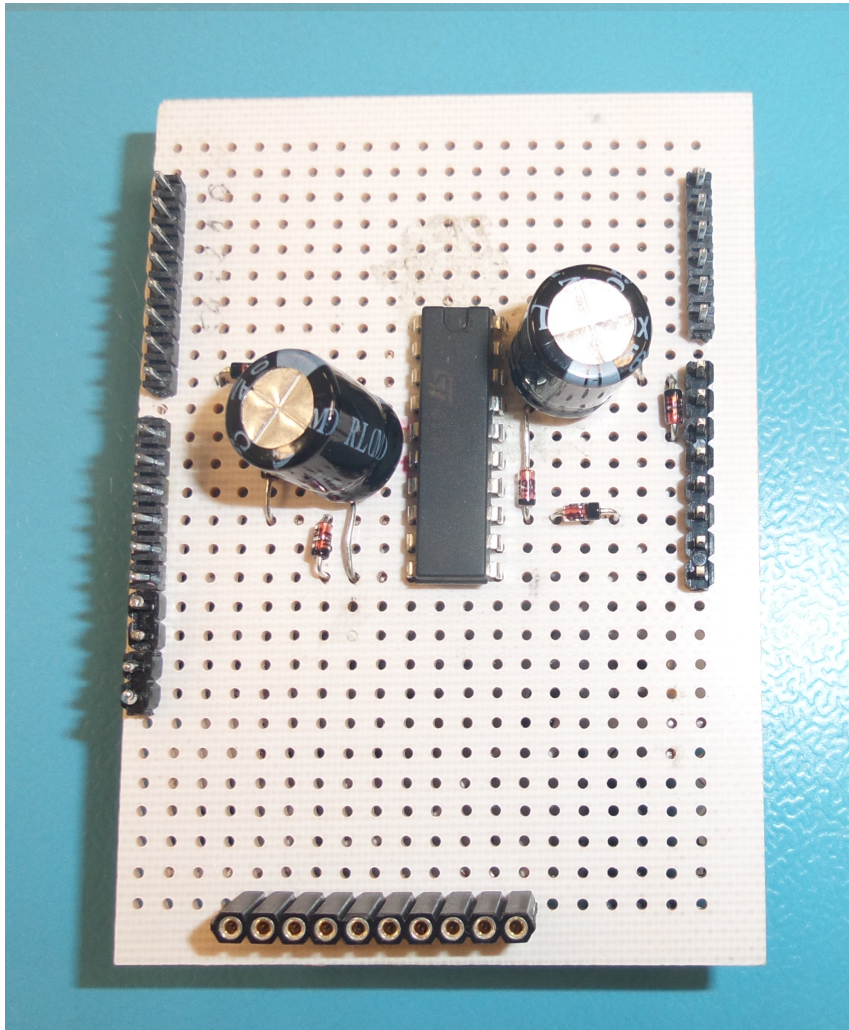
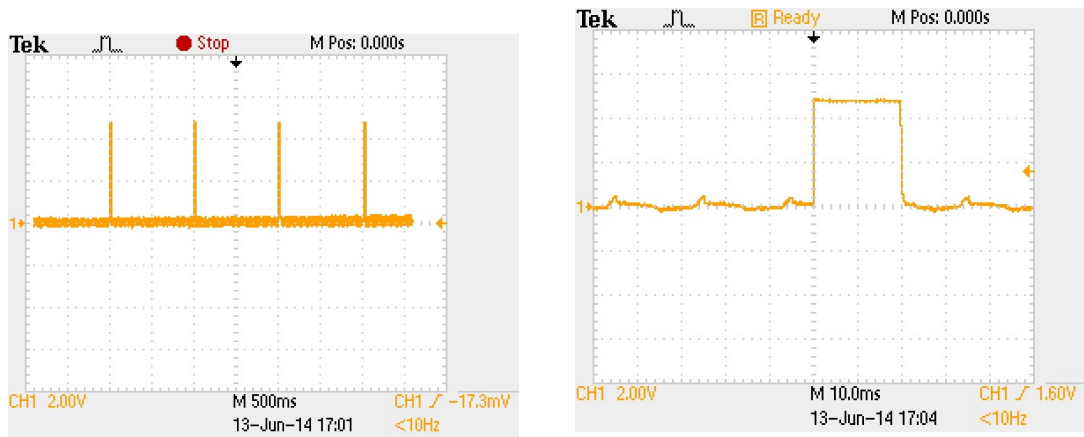


Figure 3.26: DC motor driver.

explained in chapter 5):

- Serial communication baudrate is 115200bps ;
- Number of bytes of a step request is 4;
- PWM frequency is 10KHz ;
- Time to on in the L293E IC is $1\mu\text{s}$ (datasheet);
- Time to off in the L293E IC is 450ns (datasheet);
- ATmega32u4 time to on/off is about 1μ (datasheet);

Performing some calculations, if the baudrate is 115200bps and 4 bytes have $4 * 8\text{bits} = 32$ bits, this gives about $278\mu\text{s}$ per request. Since this value is about 100 times higher than the reaction times, this is the prevailing value in the time to perform. This means that, since the request started,



(a) Waveform showing the 1s period.

(b) Waveform showing the 20ms step duration. Notice the noise.

Figure 3.27: L293E step waveform. Notice the 5V peak voltage.

the motor will only start moving in about $300\mu\text{s}$ plus the motor magnetic and mechanical reaction time.

The smallest step possible, theoretically, is one PWM cycle with the lowest duty cycle that moves the motor. The real value needs to be tested experimentally and that is presented in subsection 4.5.2.

3.2.3.2 Consumption Test

Like the PZT bender driver (see subsection 3.1.3.2), a consumption test was conducted to evaluate the time that the battery lasted with non-stop operation. The smartphone was charged up to 100% and a simple infinite loop of a full grid (10×10 steps, see section 4.5) route was coded. Each step was made with a duty cycle of 100%, 20ms duration and a 1s interval. This interval was smaller here because the bender implementation needs more time and actuations between steps. This was the way found to balance both tests.

The test was started at 20h53 and featured 4.269V and 100%. After 1h30 the test was stopped and the the values were at 3.991V and 84%. Doing a bit of extrapolation:

$$m = \frac{4.269 - 3.991}{15\%} = 0.01853 \quad (3.23)$$

$$4.269 = m * 100 + b \Rightarrow b = 2.416 \quad (3.24)$$

Like the previous test, this puts the lowest voltage at 2.4V and the minimum should be around 3V. Using only the voltage levels, this gives the function marked as red in figure 3.22a.

The plotted curves can be observed in figure 3.28. Using the voltage values and doing a simple

proportion calculation:

$$0.278V - > 1.5h \Rightarrow \frac{(4.269 - 3) * 1.5}{0.278} = 6.85h = 6h 51min \quad (3.25)$$

The value in equation 3.25 represents the approximate autonomy of this system, assuming non-stop working.

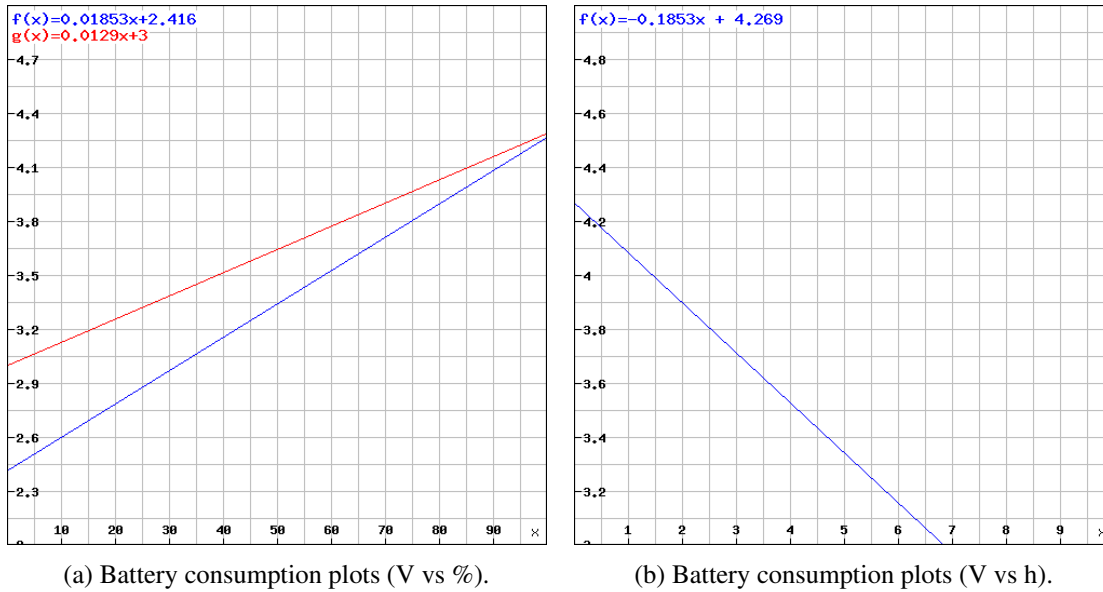


Figure 3.28: DC motor driver consumption plots.

3.2.4 Remarks

After the driver presentation, some remarks can be made:

- It works perfectly and this is one of the valid solutions to drive a DC motor. It is easy to develop and implement;
- The semi open-loop configuration is not ideal but, since the precision requirement is really loose, this is enough;
- If a current limiting method is needed, a better one should be developed. A resistor wastes precious power;
- The driver consumption was low with an estimate of $6h 51min$ (extrapolated value) of autonomy;
- The reaction time is not significant and is set by the communication between the smartphone and the Arduino;

- The cost of this implementation is 6.94€ without a PCB. If a shield like board is made, an approximate cost is 116.33€ per two boards (same price of the first driver). The full price is 65.105€. This is a prototype cost. In a large scale production, the cost can really go down. Using the PCB Pool price comparator, 100 PCBs cost 424.84€ and that makes each one cost 4.25 €, for example. Using the prices given by Mouser, if 100 drivers are made, the cost is 5.311 € per board. This totals 9.561 €. Another good idea might be to buy an already ready shield for the Arduino. This needs a market survey and will be made in the future.

3.3 Conclusion

Now that both drivers are presented, a comparison can be made between them:

- **Price** - this is the easiest comparison that can be made. Leaving the PCB out (it could be changed to cost less in both cases), the PZT driver costed 17.96€ to make while the DC motor driver costed 6.94€. The second easily wins in this field;
- **Performance** - this is really subjective since both implementations worked. Although, the DC motor driver, because is way simpler, did it flawlessly while the PZT driver had some limitations;
- **Firmware** - the software for the DC motor driver is a lot simpler than the PZT one. Is this relevant? Probably not because the firmware will be provided ready to perform. It could only influence the alteration easiness;
- **Actuation precision** - again, the DC motor driver doesn't limit the motor functionality while the PZT driver limitations can... This point is more dependent of the mechanical frame than the electronics. And, to be fair, the PZT bender can be displaced much more precisely than the DC motor moves and stops;
- **Power Consumption** - The DC motor driver wins with *6h 51min* versus *5h 28min* autonomy time. It is almost *1h 30min* more although the PZT driver autonomy is expected to be higher with a successful PCB implementation. Even if it gains the necessary time to surpass the DC motor driver, it must be for a lot (must gain over *2h*) or the difference in price can't be overlooked. But, for now, the DC motor driver clearly wins.

These general points show that the simplicity of the DC motor driver can really weight when is time to pick a final path for the system but, since the biggest reason of choosing the benders was their low power consumption, that's the most important point. This bender advantage was not noticeable in the drivers implementation because of the reasons enumerated in subsection 3.1.3. Taking out the chance of improvement of autonomy time in the future, the only advantage of the bender driver is the higher precision of it (and being a novel solution, but that may not count).

This may or may not surpass the DC motor driver advantages because it wins clearly in the price, easiness and autonomy time.

At the time of the writing and with the tests performed, the DC motor driver is probably the best option. It can even be improved with more one chip and allow more two DC motors to be used (interesting, allows the addition of a Z axis motor). But this relates only to the driver choice and the best is a bit relative since these are two viable solutions that work well for their objectives. For a full analysis and conclusion, the next chapter is needed.

Chapter 4

XY Table

This chapter is focused in the development of the physical part that constitutes the XY Table. In it, the actuation principle is explained, followed by a report about the experience of working with the 3D printer and finally presenting the various topologies pursued during the project time-frame.

Every topology is escorted by a full explanation of the principle theory and the physical structures of them are detailed. The results of the tests are presented too, making an analysis of the success/insuccess of the implementation.

4.1 Principle Theory

The method idealized is based on the stick-slip phenomenon and was already lightly presented in subsection 2.2.4.2. There is little consensus in the literary work studied (a good explanation can be found in [41]) but, generally, it's agreed that the behaviour results from the interaction of two surfaces that are influenced by thermal changes during different phases of the movement. The responsible forces are the static and kinetic friction. Normally, the static friction is higher than the kinetic one but, if a force applied is large enough to overcome it, there is a sudden jump in velocity of the movement [42].

A good example is to imagine an object (a cube for example) on top of a sheet of paper. If the sheet is moved slowly, the cube will move with it (stick) but, if there is a sudden change in the sheet velocity, the cube will stay behind (slip). If the start place is compared to the end place, we will see that the cube moved.

To sum this idea, in figure 4.1 the principle can be observed applied to the project in hands. During the slow extension (represented by II), the platform moves with the actuator (stick) because the static friction is larger than the kinetic one. Then, the actuation voltage is reversed in a very small time (represented by III) and, in consequence, the actuator reverses its movement too but, since this change is so fast, there isn't enough static friction and the slider stays in place. This traduces in a step. In the figure is also shown the control method of the actuator in terms of the voltage applied to the PZT bender. The electronic driver must be able to produce the voltage

triangles needed by the system with the correct timings. The tests were already presented in subsection 3.1.3 and they shown that the driver is valid and correct for the implementation.

This stick-slip principle is very simple and observed in numerous ways in our day-to-day life but at the same time, too hard to be described by mathematical equations (derived of the general lack of knowledge of the friction behaviour and the lack of proper instruction in mechanics). This brief presentation should be enough to try to apply it but can't be treated lightly. It is an important subject in mechanical engineering and a more knowledgeable discussion can be found in [41].

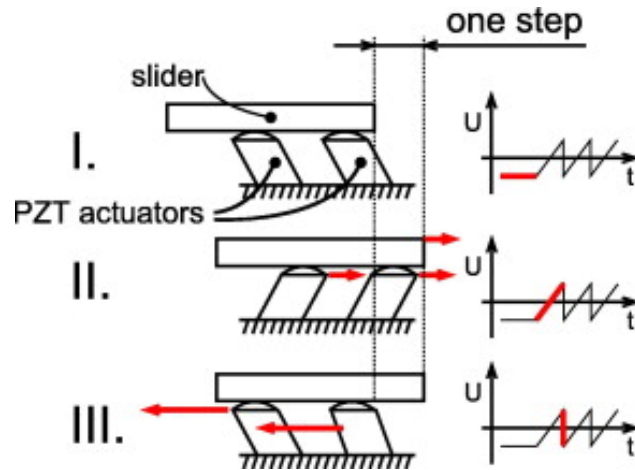


Figure 4.1: Same figure presented in 2.2.4.2, see 2.25

4.1.1 Implementation premisses

To apply this principle, the main concerns are the actuator force, the point of contact, the material of contact and the movable body. Only a correct tuning of all of those concerns leads to the wanted behaviour in an efficient and reliable way. Analysing them:

- Actuator Force and displacement** - the PZT bender chosen in subsection 3.1.1.1 features a blocking force of $2N$. This value is only achieved when the bender is in its original shape, dropping almost linearly with the displacement (see figure 2.18). This is the major point of reference of all the design. We must take in account that is almost impossible to transform all that static force into usable one. One more important reference is the fact that only half of the maximum displacement will be used (the $50V$ used, discussed in previous chapter, represents half of the possible displacement), representing half of the blocking force, $1N$. So, the upper boundary of the applied force is $2N$ and the lower one is $1N$, without any kind of amplification. As for the displacement, the logic is the same: with $100V$ the displacement is maximal and is $1mm$. So, for $50V$, the displacement is $500\mu m$;
- Point and material of contact** - probably the most important points in all the project. This will define the efficiency or even better, the success of the implementations. Analysing:

- Point of contact - this point is really topology dependent. Although all will follow almost the same premisses. Its shape should be such that the moving structure is always in contact with it. This translates in a rounded shape to compensate the rounded movement of the bender (see 2.2.3.2). Another premiss is that its shape will be reflected according to an axis to allow movement forward and backward. This will be explored in each topology;
- Material of contact - this a really difficult point. The material used needs to provide enough friction to the moving structure for it to stick but, during the retraction, the friction needs to be low enough that the structure stays in place or, at least, the backward displacement is lower than the forward one. For this point, a web search was performed and a good information source was found [43]. In it, we can observe that there are a number of materials that present high friction coefficients like Aluminium, Glass, Iron or Rubber. Of all, the easiest and cheapest one to find is the rubber. Since there are thousands of types of rubbers the most available ones were studied (see 4.2):
 - * Common white rubber - this type is not hard enough, it bends too easily and can't transfer the force;
 - * Common green rubber - this one is really good for the application. Easy to cut, model and hard enough to apply force. This type was chosen;
 - * Mixed rubber, red part - better than the white type in terms of bending but not enough;
 - * Mixed rubber, blue part - as good as the green rubber but harder to cut;
 - * Isolation rubber - not suitable for the application, extremely hard to make the forms wanted.
- **Movable body** - it should be as light as possible and its point of contact with the actuator will be lined with the same material used in it. If this provides too much friction, it will be removed. It will be printed in the 3D printer so it will be made of PLA or ABS (see section 4.2). This will need to be fixated to some kind of structure so the friction with the point of contact needs be really low. Some mechanical aids (ball bearing for example) might be used depending on the success/insuccess of the topology.

Everything presented here should be taken as an introductory and preliminary analysis to the problem. As will be shown in the rest of the chapter, some ideas and premisses changed and evolved during the development derived of each small step and topology iteration and are properly explained.

4.2 3D printer

The 3D printer used is a Makerbot Replicator 2x. This printer is in an experimental phase and uses the principle of plastic extrusion where a type of it is heated and then pushed through an



Figure 4.2: Studied rubbers.

extrusion nozzle head. It immediately hardens and form layers. This process is controlled by a microcontroller that turns the flow on and off and moves the head to the correct point.

This specific model features a dual-extrusion mechanism that allows the simultaneous extrusion of different materials. Makerbot claims that it features $100\mu m$ of layer resolution, $11\mu m$ in XY and 2.5μ in Z. These are impressive values and should be good for the project.

The software used can be the Markerware (property of Markerbot) that works with any STL 3D model. There are some open source variants of it are more flexible (and bug free) like Replicator G.

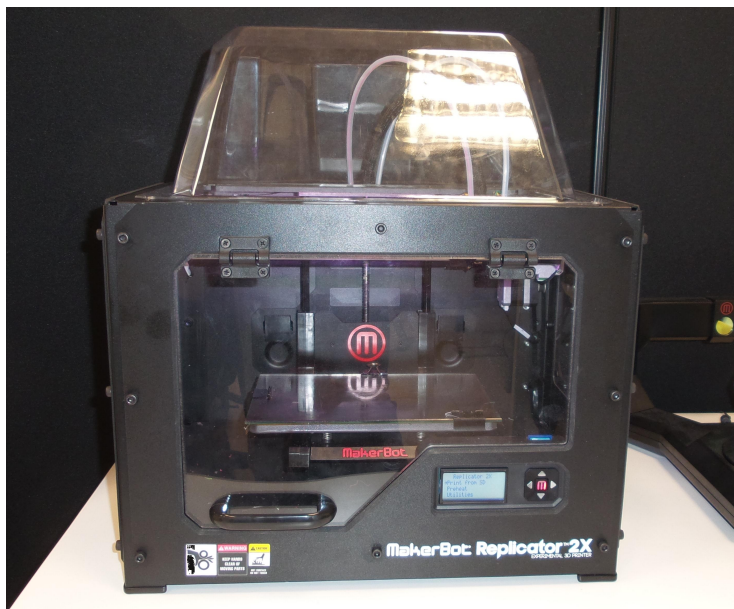


Figure 4.3: Makerbot Replicator 2x.

4.2.1 Materials

The two main materials used with the Replicator are the ABS and the PLA. As explained above, both are thermoplastics (soft and moldable when heated, solid when cooled). Both are valid options for 3D printing but some characteristics change and that's the reason why some people prefer one over the other:

- **Storage:**

- ABS - If there is some humidity in the place where it is stored, it will tend to create bubbles and spurt during the print;
- PLA - Same problems of ABS but has an even greater negative influence with alterations in extrusion temperature.

- **Smell:**

- ABS - Smells like hot plastic (may be a problem in closed environments);
- PLA - Since PLA is made from sugar, it has a sweet scent that is better than hot plastic.

- **Accuracy:**

- ABS - Strong and somewhat flexible material. Is resistant to high temperatures and is often the preferred plastic for projects with mechanical uses in mind. Corners present a slight rounding;
- PLA - More rigid than ABS making it more difficult to interconnect different pieces. This material has tendency to curl if the build plate is not hot enough. If actively cooled, much sharper details can be achieved.

A more extensive discussed can be found in [44].

Analysing the presented pros and cons, the ABS characteristics are more attractive for the structure in mind because there will be a lot of interconnection pieces (no screws requirement). The somewhat flexibility of the material is the big advantage over the other material. The ABS used was Markerbot's that featured a filament diameter of 1.75mm.

4.2.2 Calibration and Tests

Before any printing can be done, the printer needs to be calibrated. This is achieved levelling the build plate, using the existing screws. This process is made with the help of a cardboard that should be able to move under the extruder with a bit of friction.

The first few prints (a simple cube) were not successful (4.5). They were printed with the Markerware default settings using ABS. They had lot of flaws and its corners curled up.

After some days of more testing, a new glass base was inserted into the printer (4.6). Around the web, there were reports of improvement with the parts and the curling. The quality of the new pieces improved indeed but they were nowhere near the promised resolution by Makerware.

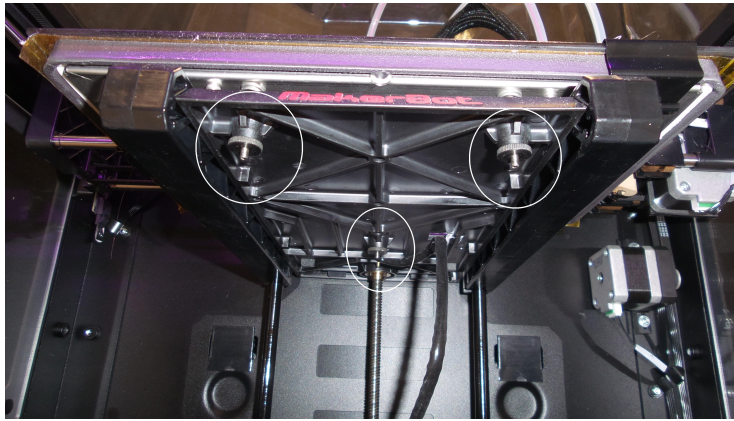


Figure 4.4: Build plate spring screws.

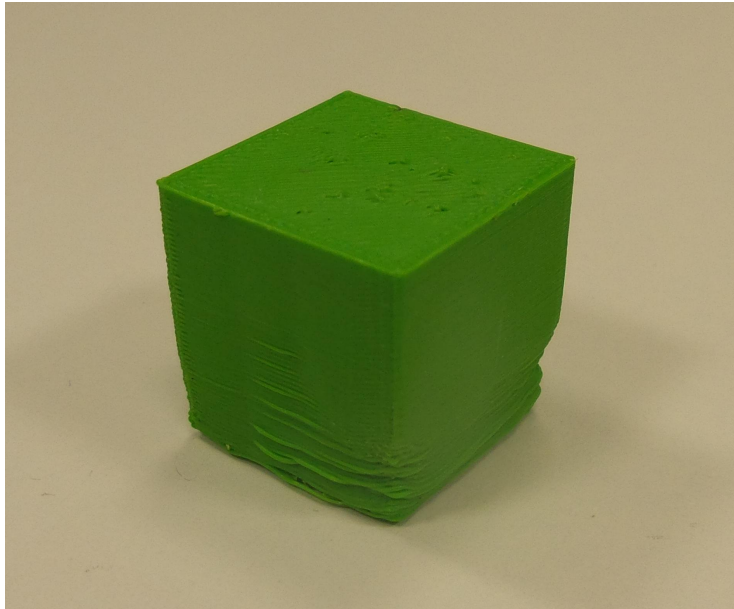


Figure 4.5: Failed cube print.

There were errors in the scale of millimetres. The Replicator G software was tested too and its settings changed. A comparison between the failed cube and another printed with the glass base and Replicator G can be observed in figure 4.7.

The best setting found was using the Replicator G default settings but with the built plate at 130°C . This presented reasonable results but, again, nowhere near of those expected (this can be testified by the previous users of the printer). This conditioned the implemented topologies as we will see next.

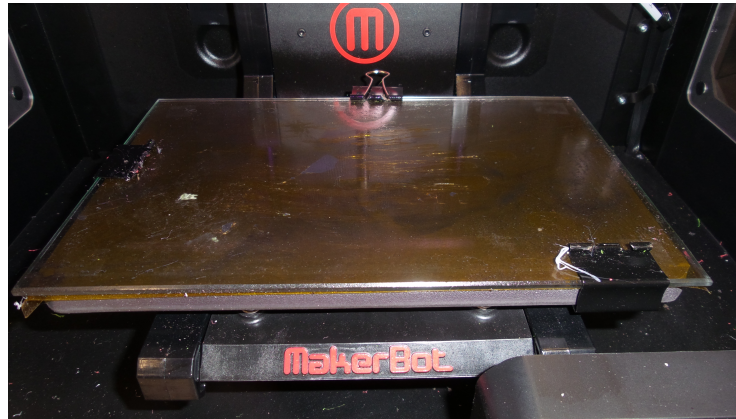


Figure 4.6: New glass base.

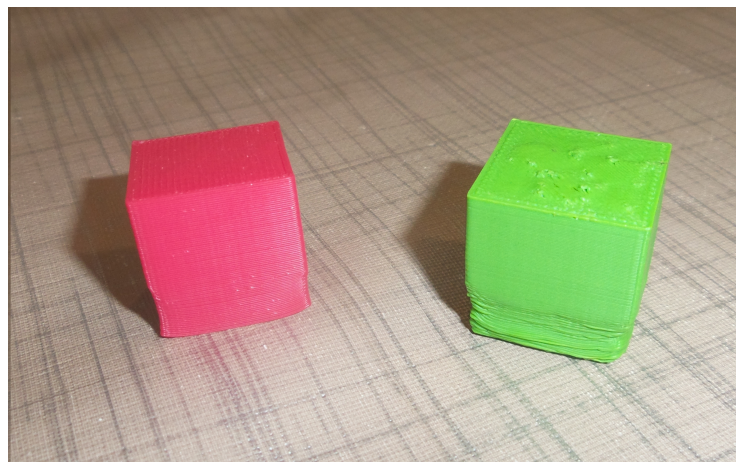


Figure 4.7: Cube comparison. Notice the curled bases and the improved resolution of the red cube.

4.3 Triangular Tip topology

4.3.1 Model Development

This solution was developed with flexibility in mind, trying to allow several uses for the motor. Its 3D model can be observed in figure 4.8. Everything is in scale with the bender that features its measures as detailed in the datasheet.

Analysing it, some of the most important parts are the ones that belong to the clamping mechanism of the PZT bender and they are marked with 1 in figure 4.8. The complementary pieces clamp the bender (black figure) and fit in the main body of the motor. The front cover then keeps everything in place and features a hole to allow exterior observation of the mechanism.

The tip is marked with the number 2. It is the most important part of this construction and features a slot to insert the actuator. This slot allows the use of the force to the both sides equitably. As observed, the tip has a hole where an axis will be inserted. This axis needs to be robust and the choice fallen into a paper clip part. It is easy to obtain and strong enough to hold in place. As

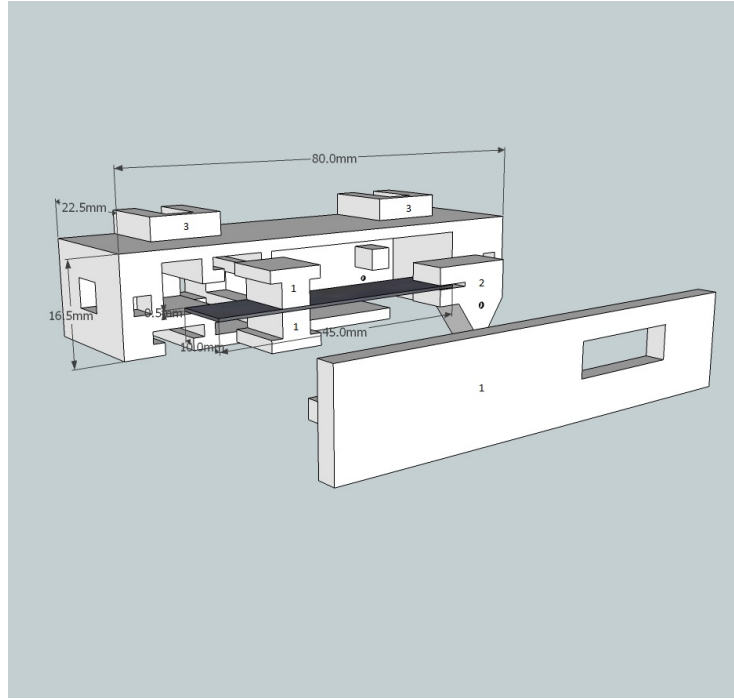


Figure 4.8: Google Sketchup 3D model of the triangular tip topology motor.

for the dimensions, they were obtained by making an analysis over the mechanics of the simple lever machine created that can be observed in figure 4.9.

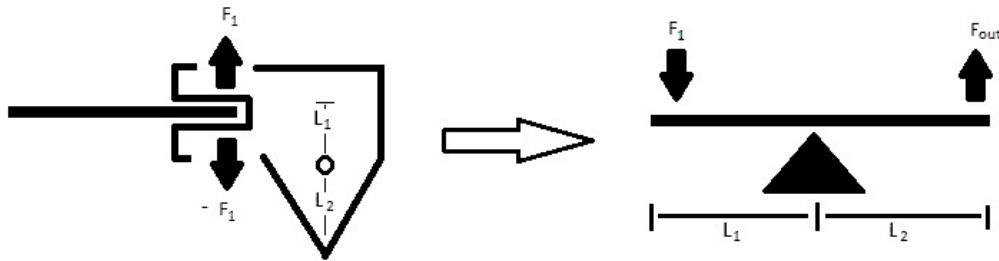


Figure 4.9: Lever theory applied to the triangular tip topology.

The force and displacement ratings were discussed previously and are $1N$ and $500\mu mm$. In the lever theory, $\frac{F_{out}}{F_1} = \frac{L_1}{L_2}$ and that means that we can achieve force amplification only playing with the lengths. The downside of this method is that the displacement will lower by the same amount of the force increase. Since no amplification is wanted because we are producing movement in steps and the material will be really light (1.04 grams per cubic centimeter), L_1 and L_2 were projected to be equal and measure $5mm$.

The tip will work as explained in section 4.1 and will be coated with the rubber discussed before. The movement of the tip is considered horizontal because, since it will be coated with the

rubber, it will provide the necessary support to the axis.

The supports marked with 3, are used to provide fixation to the motor. The slots in the bottom part are used to provide fixation to the movable axis. Finally, the hole in the side of the motor was inserted because of the bender wires.

The axis supports were projected in a way that the axis will always be in contact with the tip. Here lays the biggest need in precision.

The full (one axis) system can be observed in figure 4.10. To conclude, everything was developed with the smallest possible dimensions without compromising robustness.

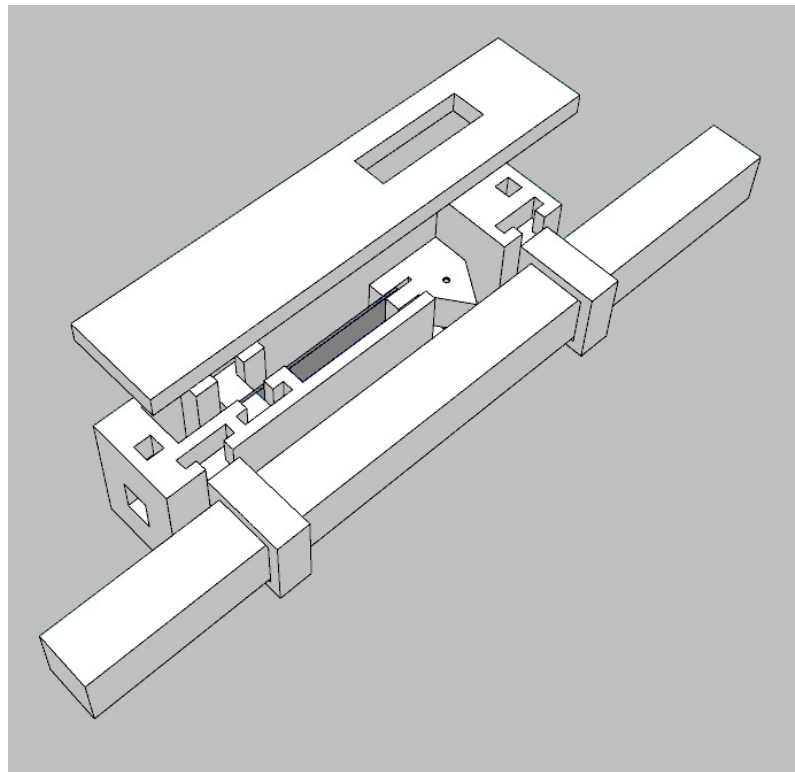


Figure 4.10: Google Sketchup 3D model of the triangular tip topology.

4.3.2 Physical Implementation

After the 3D models ready and confirmed, it was time to print the pieces. As said above, the software used was the Replicator G with default settings (with some changes) that are:

- Object infill = 10%;
- Layer Height = 0.27mm;
- Number of shells = 1;
- Feedrate = 40mm/s;

- Travel Feedrate = 55;
- Print Temperature = 230°C
- Filament Diameter = 1.75mm
- Nozzle Diameter = 0.4mm
- Build Plate Temperature = 130°C

All the necessary pieces for testing an axis were printed and the best ones can be observed in figure 4.11.

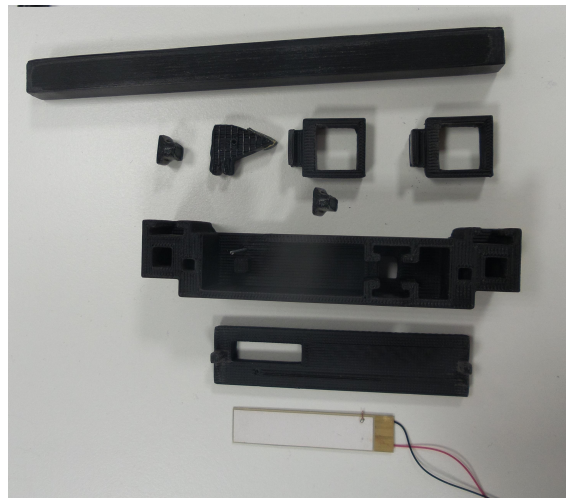


Figure 4.11: Triangular Tip topology printed pieces.

This pieces were really far from perfect, not a single piece fitted in their socket without some sanding and polishing. All presented some curling and had their dimensions with errors of millimeters. To give an example, we fully measured a random single piece to perform some comparisons. The piece is in figure 4.12 and the measurements in table 4.1. equal

Measure	Expected (mm)	Real (mm)
1	19	20.65
2	14	15.23
3	2	1.94
4	2.5	4.09
5	2	2.26
6	2	2.03

Table 4.1: Measurements comparison.

As shown, the resolution of the printer is far from what is advertised by the manufacturer. This test only complied with the one shown in subsection 4.2.2.

Even with the problems shown, the system was assembled and can be observed in figure 4.13.

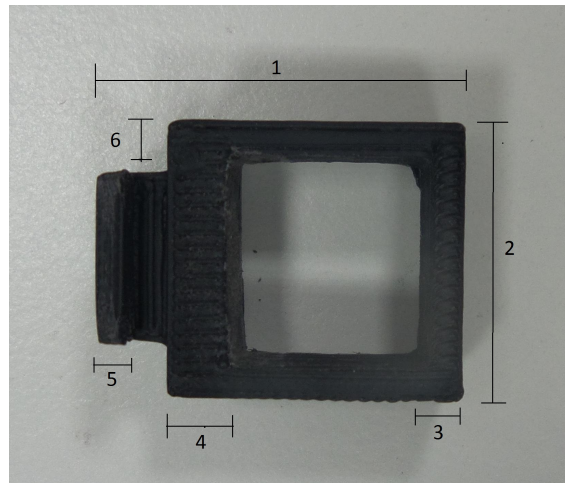


Figure 4.12: Sample printed piece.

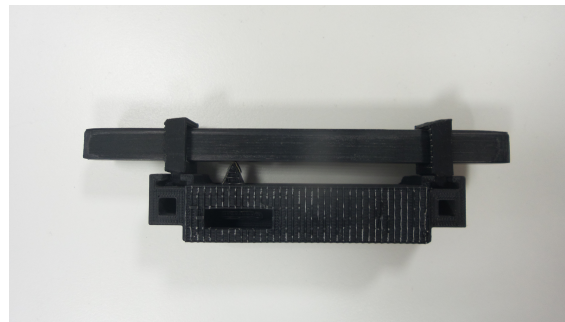


Figure 4.13: Triangular tip topology assembled system.

4.3.3 Experimental Results

Derived of the lack of precision, the structure was loose and the axis was not hold firmly. This was already a fail but the tests continued. The voltage wave shown in figure 4.1 and discussed in section 4.1, with limits in $+50V$ and $-50V$, was applied to the actuator with a period 10000 times superior to the $10KHz$ used by the driver ($1Hz$). This value was used because the publication [42] used it with success and this makes it a good starting place. The wave produced can be observed in figure 4.14. There are two phases, one when the voltage rises slowly to the side of the movement and another with a rapid contraction to the lowest value of the voltage.

It should be noted too that the waveform was fully controlled by the smartphone using the library that will be presented in chapter 5.

As expected, the topology didn't work. The axis was not hold firmly enough and that caused it to leave the contact with the tip. This was compensated with more rubber in the tip but all this was just a patche to the bigger problem, although some movement was achieved. This movement was just stick movement, the slip part wasn't achieved making it useless. With more tries in patching the problem, the desired movement was never produced. A robust system was wanted and this

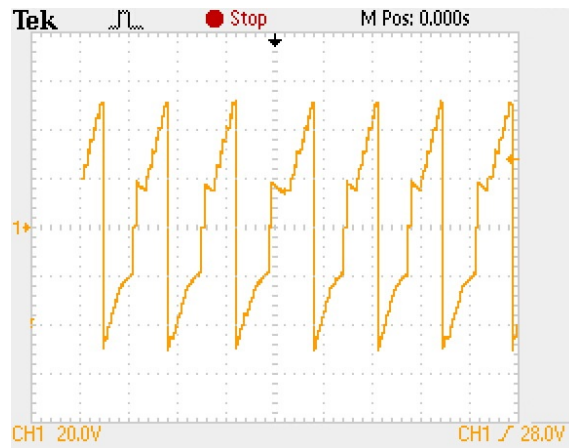


Figure 4.14: Applied voltage to the actuator. Note the dead zone of the converter.

wasn't nowhere near robustness.

4.3.4 Remarks

To conclude, some remarks can be made about the topology:

- All the pieces developed are printable;
- No good precision can be expected from the available 3D printer. This was proven by the numerous tries;
- The driver can produce the necessary waveform without any problem. The used frequency must always take into account the 10KHz frequency of the driver;
- The system must be less precision dependent. This will require a careful rethinking of the system;
- This idea should not be abandoned. It could work if the pieces are printed well enough.

After those remarks, some decisions were taken:

- Before trying to print everything, the concept should be proved. For that, a good working mechanical system will be used as base for the motor actuation;
- The system should be manually adjustable. This will allow a deeper test of everything;
- After successful testing, then the rest of the structure should be printed.

With everything in mind, a new topology was sketched and tested. It is presented next.

4.4 Claw/Lever Tip topology

4.4.1 Model Development

As said before, the focus was in proving the concept. For that, a proven mechanical system was needed to remove the need to print the axis system. Using the research made in section 2.5, two solutions were found:

- Igus Drylin T miniature - these systems are made for applications like the one studied for this project. They are excellent to test the concept but, probably, won't be used in the final system because they are expensive (see prices in section 2.5);
- CD-rom drive axis system - this is a way cheaper solution than the previous one but not as good. Might be a solution for the final system if the axis can't be printed.

With those two hypothesis in mind, Igus was contacted and a meeting was arranged with Ing. João Poças (product manager). After a brief exchange of ideas, he offered his help and supplied some samples of the Drylin systems to be used in the project. They can be observed in figure 4.15.



Figure 4.15: Igus drylin T miniature system.

With this problem solved, it was time to develop the 3D model of the actuation system. Some ideas were in mind while developing it:

- The system must be adjustable;
- The system must allow several different tests;
- The system objective will be to move the Igus mechanical platform.

The system developed can be observed in 4.16. It features a clamping part recycled from the previous topology with some adjustable supports. The idea is to use screws to hold it against some

base (wooden base for example) but provide space for adjustments. Then, there is another part that translates the bender movement to usable force. It features a changeable tip, hold in place by a base where an axis (paper clip) is inserted into a round slot and hold in place by a top cover. This base has the same type of supports of the clamping shell to allow manual adjustment.

The tip is changeable. This allows several different tests of actuation without needing to change the whole system. For now, two tips were developed. Both used the principle of levers, discussed in subsection 4.3.1, in its development but work in different ways. Analysing them:

- The tip numbered with 1 amplifies the displacement using a 2 : 1 ratio in the lever. This makes the displacement 1mm for each side. Then, applies the stick-slip method to the moving platform using the rubbers glued to it (represented by the green triangles). The rubber's shape is not pointy but sanded, creating a round point of contact;
- The tip numbered with 2 amplifies the force using a 2 : 1 ratio in the lever. It applies the stick-slip method directly with the rubber. The rubber's tip is rounded too.

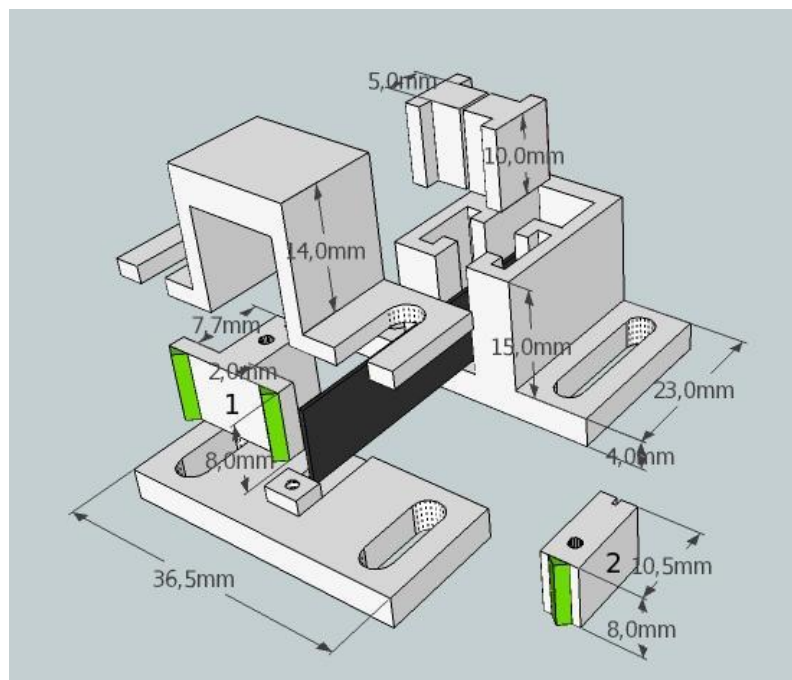


Figure 4.16: Google Sketchup 3D model of the claw/lever tip topology actuation system.

The full system with tip 1 assembled can be observed in figure 4.17. Some supports were created to hold the Igus system, as seen in the picture. Another important fact is that the supports were made with M4 screws in mind. This can be changed easily.

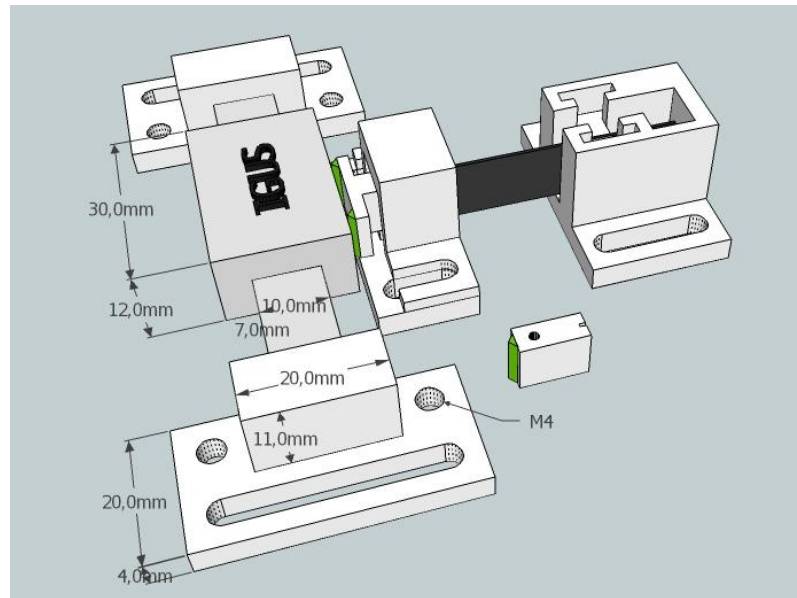


Figure 4.17: Google Sketchup 3D model of the claw/lever topology using an Igus system model.

4.4.2 Physical Implementation

All the pieces were printed with the same settings used in subsection 4.3.2. Like the previous topology, the pieces didn't fit together without some sanding. This was expected because of the previous example and the resolution was almost the same.

With all pieces printed, the system was assembled in a wooden board and manually adjusted. It can be observed in figure 4.18.

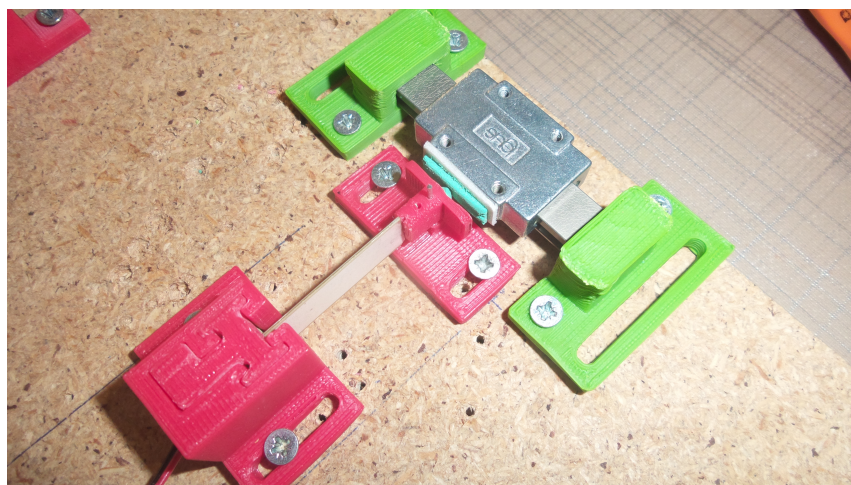


Figure 4.18: Claw/lever topology assembled system.

4.4.3 Experimental Results

The first tip tested was the number 1. This system works the same way than the others but the applied wave is slightly different. The wave has the same form of the one shown in figure 4.1 but its limits are 0V and $\pm 50V$ depending of the direction of the movement, being 0V the position of rest. The applied wave can be observed in figure 4.19 (again, controlled by the smartphone).

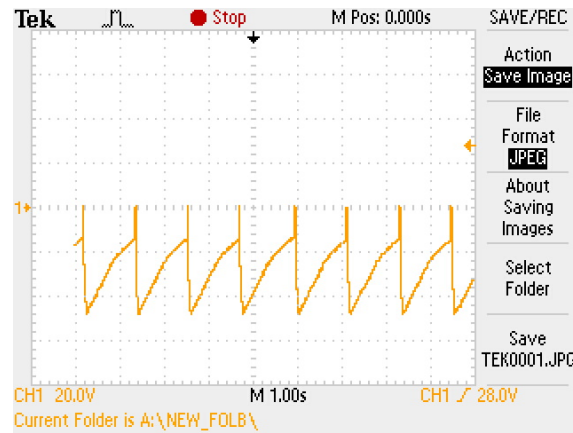


Figure 4.19: Applied voltage to the actuator.

This tip was unable to produce any kind of movement. The displacement amplification was too much and that severely limited the applied force. Even with numerous manual adjustments (moving the tip closer or farther) of the tip or changes in frequency (used 10Hz, 50Hz and 100Hz) no success was obtained.

The second tip uses the same waveform of the first topology (see figure 4.14). This time, the platform moved but ultimately stayed in place. This means that only the stick principle was in operation. With some manual adjustments, some steps were achieved but they were too inconsistent (sometimes, a step occurred) to even consider this a success although, this was promising.

4.4.4 Remarks

There are some ideas to retain from this experience:

- The first tip was a fail. The idea was abandoned;
- The second tip was able to move the platform. It presented promising results;
- The force applied is more important than the displacement. This was obvious when the second solution was experimented;

Since the time to the project end was rapidly approaching, this solution (with the second tip) was put in standby. It is indeed possible to develop an actuator with a stick-slip mechanism as seen in [42] but needs a lot more mechanical study to apply it. This might not be possible and compatible with the limitations of the project. With that in mind, two new paths were sketched:

- Develop a more simple approach with three benders. Not ideal because the design will become more expensive but good enough for a first prototype;
- Develop a solution based in common motors. This idea was good in more than one way: it allows a comparison between solutions and is easy enough with a lot of examples in literature.

These two paths are presented next.

4.5 DC motor table topology

Derived from the remarks show in subsection 4.4.4, a new topology based in common motors was developed. To develop this system without spending too much time, we recurred to the mechanical research made in section 2.5 and selected the CD-rom system to be the base. This choice was made because it is really easy to get two (one per axis) broken systems from any computers store and this goes into the thesis objective of spending the least money possible. We can also find a great variety in terms of stepper or DC motors. A DC motor system was chosen because it is easier to control (only one H-bridge per motor vs two bridges per stepper motor), consumes less power and is cheaper. The downside is that it isn't as precise as the stepper motor but, since the final system doesn't need incredible precision, we can get away with it.

This is not the final idea for the problem but rather a fast solution. Ideally, the whole system will be printed by the 3D printer and then we fit the DC motors in it. It will be just dependent of the DC motors and not of the CD-rom systems.

The used system can be observed in figure 4.20.

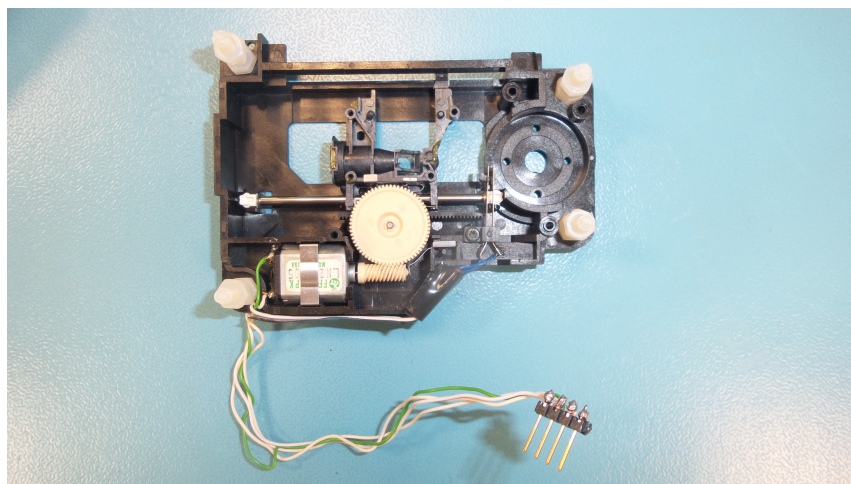


Figure 4.20: Example cd-rom system.

This particular system uses a gear to translate the rotational movement of the motor in linear motion. It is not great but does the job well. Further when the motor stops, it has no need to continue to be energized because the mechanical helicoidal transmission holds the position fairly

very well. Features a sensor placed in the end (or beginning) of the axis that is nothing more than two pieces of metal that touch each other when the platform reaches them. Only one of the sides of the platform is held by a metal axis, the other side is just lodged in a plastic support.

The motor is a Mabuchi *FF – 030PK*. This is a really small motor that features a stall current around $0.5A$ and has a nominal voltage value of $5V$. This motor starts spinning (without any load) at about $0.7V$ and is capable of moving the system at about $1.2V$. This means that the full $5V$ of the driver won't be needed and the motor can be controlled by PWM (see subsection 3.2.2.3).

One interesting addition is the use of two mechanical actuated sensors. They were already in the design, they weren't added. They signalize when the platform reaches them by creating a short circuit between two pieces of metal. This can be used by the Arduino to signalize the origin (point $(0,0)$) of the grid.

Two equal systems were used for each axis. This is ideal but not a requisite for the system. The control can be adjusted for each motor and system independently.

To conclude, there is the need to talk about the grid implementation. For a successful analysis, the system needs one-hundred images and the idea of performing a grid type movement was already refereed in section 1.2 (the movement complies with the methodology proposed by WHO for malaria microscope analysis [13]). This movement will be performed as shown in figure 4.21 using the sensors for the $(0,0)$.

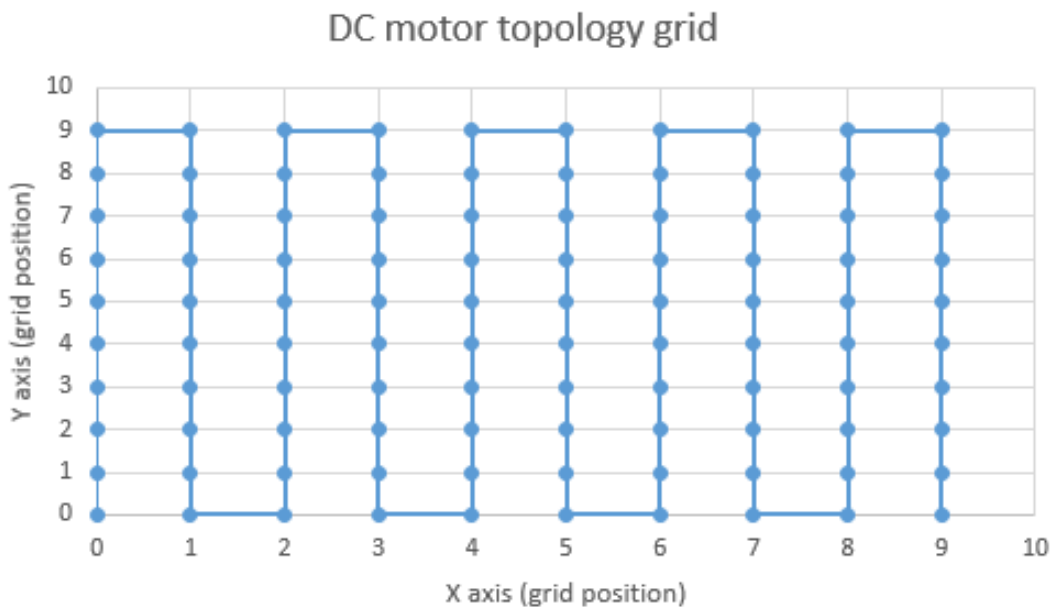


Figure 4.21: Grid movement implementation for the DC motor topology. The values represent a cell position.

4.5.1 Physical Implementation

The assembly of the system was made with the most available components in the laboratory. This design is not optimized it was just made in a way that the assembly was as fast as possible because the time was short.

Since both systems contain everything essential to work and just needed support, this was addressed. The assembled system can be observed in figure 4.22.

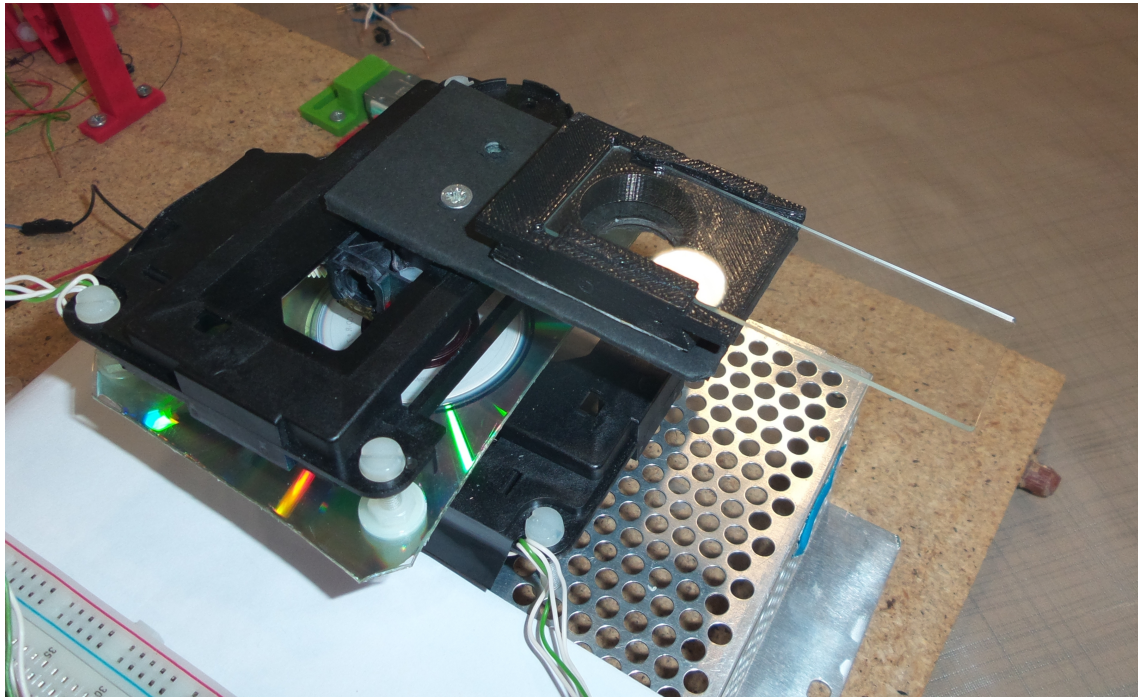


Figure 4.22: DC motor XY Table with a microscope slide.

An aluminium base was used to hold everything in place. This base has several $4mm$ holes and that was useful to hold the system in place. Some nylon $M4$ screws were used to place the X-axis system in place together with some nuts. As shown in figure 4.20, there was no obvious way to hold the other axis. For that, a support was developed that could fit in the platform without destroying it. That support can be observed in figure 4.23.

It is a simple geometric form, adjusted to the platform in hands, leaving $2mm$ free counting from the higher point of the platform. This is not ideal because needs personalization for each system and a new solution must be achieved in the future but, for now, is enough. Since the resolution wasn't a big problem with this support, the printed version was reasonable and more than enough. Two holes were made to hold it in the platform with the help of some screws and nuts. Then, on top of it, a platform made out of a cut cd-rom was fixated with a screw. It was cut with the shape of the system, leaving the hole in the middle directly under the moving platform. This could be useful for the illumination of the microscope slide.

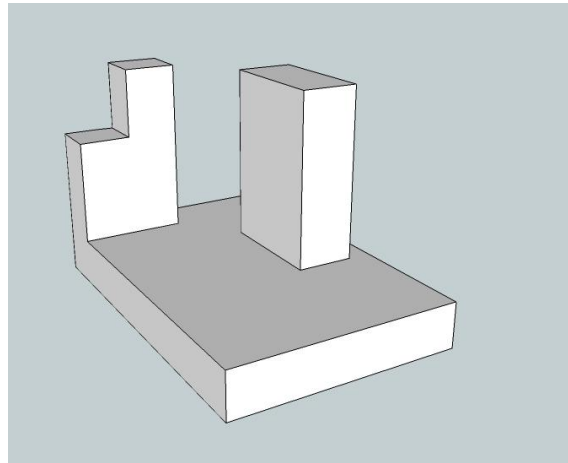


Figure 4.23: Google Sketchup model of Y Axis support.

Four holes were made in the cd platform to allow the hold of the second axis with some nuts and screws (nylon *M4*). Other support (like the one in figure 4.23) was assembled in the other axis to allow the assembly of the microscope slide holder.

In the end, the microscope slide holder was assembled and the table was ready.

4.5.2 Experimental Results

After everything assembled, some tests were performed. Some things needed testing: the validity of the table and the real resolution of it. The tests were:

- **Validity** - test if a full run can be achieved in some specified constraints, with a relaxed step size but under 1 *mm*;
- **Resolution** - test the real maximum resolution of each step for each axis with an accurate test and check if the result is under the needed step size;

The smartphone was used to control the motors and conduce the testing with the test application developed (see chapter 5).

4.5.2.1 Validity

For this validity test (and to the resolution one), a system for measurement must be used. In the laboratory, there were no machinery for this kind of application, so, there was the need for an ingenious and accurate approach with available materials.

One of the most precise items that are present in almost all homes is the optical mouse. It features a light-emitting diode and a photo-sensor to detect movement relative to a surface. A normal mouse has around 1000*DPI* (or 39.3701 *DPmm*) that translates to dots (or pixels in case of a computer screen) per inch (millimeter). This is already high, meaning that it can detect movements of $\frac{1 \text{ mm}}{39.3701 \text{ dots}} = 25.4 \mu\text{m}$ but this is theoretically because several other factors need to

be taken into account (USB delay, movement detection algorithms, etc) turning the mouse in an accurate but not precise system. Although, this will be enough to have an idea of what is needed (around $500\mu m$ steps) in a simple and cheap way. Every result should be taken with a grain of salt because the mouse lens is very susceptible to dust and imperfections so an error of $100\mu m$ is expected.

This method is not new and a full analysis can be observed in literature [45].

A computer software is needed to process the information from the mouse. Instead of developing a software from scratch (that would take a lot of time), a freeware named Mousotron was used to count the pixels movement. This software is for Windows only and uses the Mouse API to give the movement details.

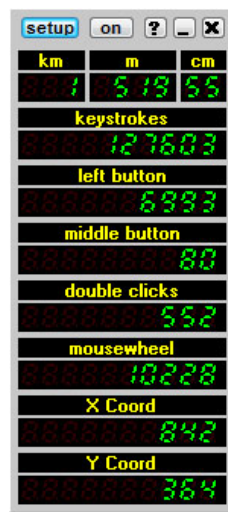


Figure 4.24: Mousotron screenshot. The "X Coord" and "Y Coord" are important fields.

To perform the analysis, the mouse used was a *X7XL – 750BK* and it features 3600 DPI (141.7323 DPmm). The system shown in figure 4.25 was assembled to perform the tests. The assembly was not very robust so it was reinforced by some electrician tape. The cardboard was assembled as parallel as possible to the mouse. This is not a perfect testbench but it is enough to give some notion of the steps.

For the validity, a run was made from position (0,0) to (2,9). The configurations used for the system were:

- Voltage applied = $5V$;
- Period of actuation = $20ms$;
- PWM duty cycle = 100% (or 1023 in Arduino PWM value) meaning that the motors were being voltage controlled;
- PWM frequency = $10KHz$ (this value doesn't matter since the duty cycle is 100%).

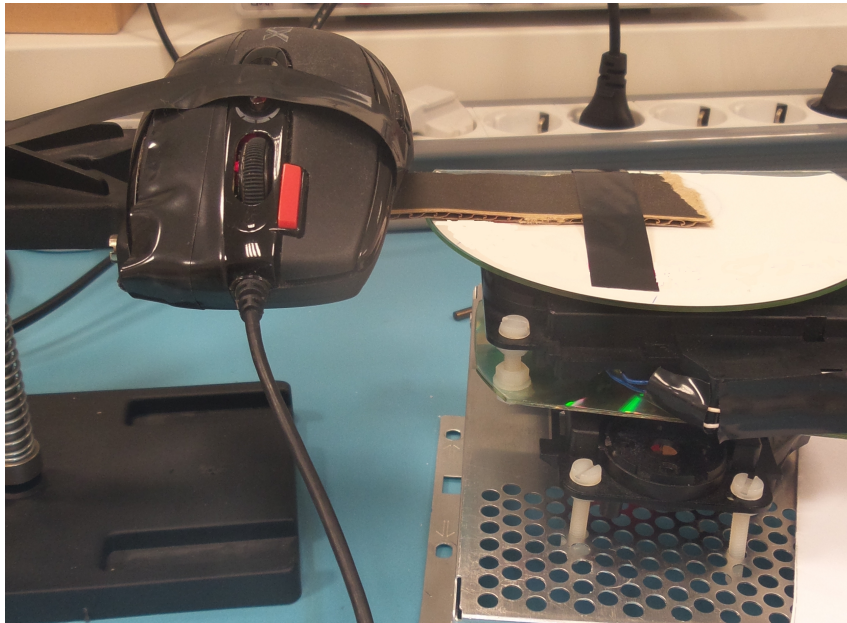


Figure 4.25: System assembled to measure the DC motor topology displacement. A black cardboard was used to enable the measure.

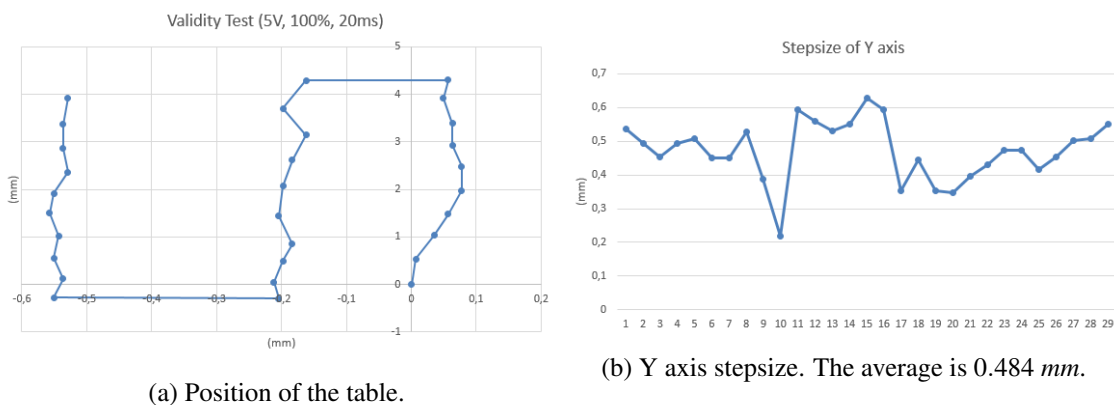


Figure 4.26: DC motor topology validity test.

The results can be observed in figure 4.26.

As shown in the previous pictures, the measure system works very well. It induces some error as expected but does its job. About the system, it can be observed that, even with some relaxed configurations, the step size is about $500\mu m$. That alone could be the final step size because it is enough for the 10×10 grid in a $20cm$ diameter sample.

There was not enough data to characterize the X axis. It will be done in the resolution test. Probably, both axis will need different configurations because the X axis has more weight to move.

In the figure can be observed the error from the mouse. The extra back-travel is physically impossible because the system started in maximum available position. The lack of repeatability can be observed but that can be error from the measure system. Although, the performance was

really good and shown that is possible to draw the grid in an open-loop configuration.

4.5.2.2 Resolution

For the resolution test, it was just a case in reducing the PWM duty cycle and period of actuation until the motors stop moving. When that point was achieved, the configurations were slightly increased and the values were defined. The method of measurement was the same used in subsection 4.5.2.1.

The greatest problem found was the inability of the measure system to read steps with the step time lower than $5ms$ accurately (better, with the same tolerance than the validity test). This was the wall achieved and the results can be found in figure 4.27. The step sizes of each axis are shown in figure 4.28.

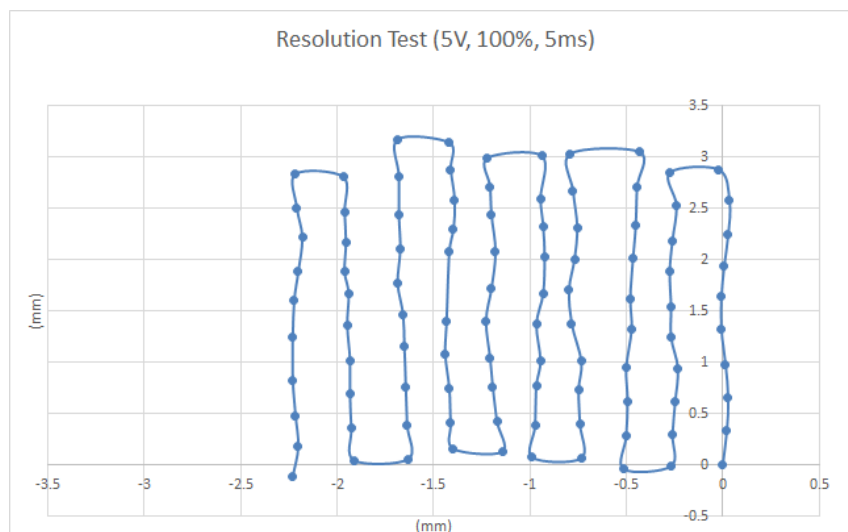


Figure 4.27: Position of the DC motor table.

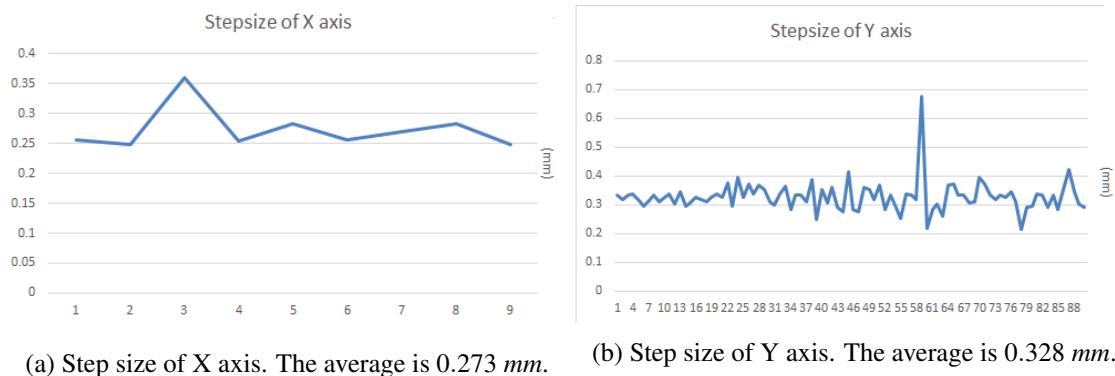


Figure 4.28: DC motor topology resolution test step sizes.

These values took a lot of time to take. Several tries were needed and even more adjustments were made. It was really hard assembling the measure system in a way that the values were always reliable and without a lot of noise.

The values shown represent a really good resolution. This time, the values seem more stable than in the validity test. That might be a result of a better assembly or a cleaner platform of movement. As expected, we can see a difference of resolution in both axis derived of the extra weight that the X axis must move. The lower number of comparison points can perform a role in the difference too. It must be said that lower step times are possible but the measuring system just can't measure them. This means that even more resolution can be achieved but a new, more professional, measure system is needed.

It cannot be said with 100% certainty that the system has a resolution lower than $300\mu m$ but everything points to that. Even with only $500\mu m$ resolution, this system is able to respond to the proposed task.

4.5.3 Remarks

Summing up the DC motor topology, some remarks can be made:

- This system is very simple and does exactly what is needed;
- The use of old CD-rom systems was a good idea but is not viable for the final system. A new model must be developed around the DC motors;
- This system is capable of under $300\mu m$ steps (needs further testing), more than enough for the project;
- Overall, the system is very cheap and easy to implement. This might be the best path for the future of the project.

4.6 Three bender table topology

Derived from the remarks show in subsection 4.4.4, a new topology based in three benders was developed. Since the main problem was the slip phase of the actuation principle, a third bender was inserted to remove the contact from the actuators and the moving platform during that phase.

This is a patch of the intended topology, to prove that a PZT XY Table can be developed and to show a possible solution.

4.6.1 Model Development

The model developed uses the friction forces to provide the intended movement. There are three benders, one for each axis and a complimentary one to do the function explained before. This complimentary bender needs a few electronic components to work. To the output of the PZT driver, the circuit shown in figure 4.29 was added.

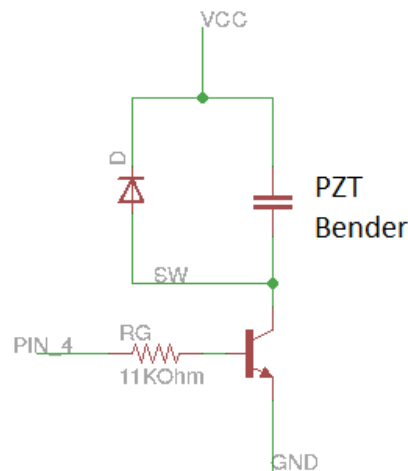


Figure 4.29: Third bender circuit.

This topology arrangement can be observed in figure 4.30.

As show in the picture, this system is based in a simple cd-rom. This decision was taken because they are really common everywhere and are a cheap way to get some flat plastic. This cd can even be a printed platform, this is really flexible. The form of the platform can even be changed if desired. It is supported by four printed supports to elevate the system. The axis actuators use the same clamping system discussed in section 4.3 with a modified tip (see figure 4.31). The tip is a rounded piece of rubber that guarantees that it is always in contact with the moving platform and it keeps the 2:1 amplification system discussed in the lever topology.

Those two cubic forms are just a way to limit the travel of the system. The distance is more than enough for the 20 cm travel needed.

This arrangement is more a $X\theta$ table. This is a possible solution because there is no need of a precise XY table, just 100 different positions and this solution produces that easily.

The idea is that the actuator numbered with 3 is like a circumference center for the theta movement originated by the actuator numbered with 2. This movement follows a simple procedure:

1. The actuator numbered with 1 and 2 start at rest (0V);
2. Actuator 1 goes up (simply by applying the +V);
3. Actuator 2 goes to the opposite side of the movement (+/- V);
4. Actuator 1 goes down (0V);
5. Actuator 2 goes to the side of the desired movement using the wave show in the point II in figure 4.1 (maximum value is the contrary of the one presented in 3).

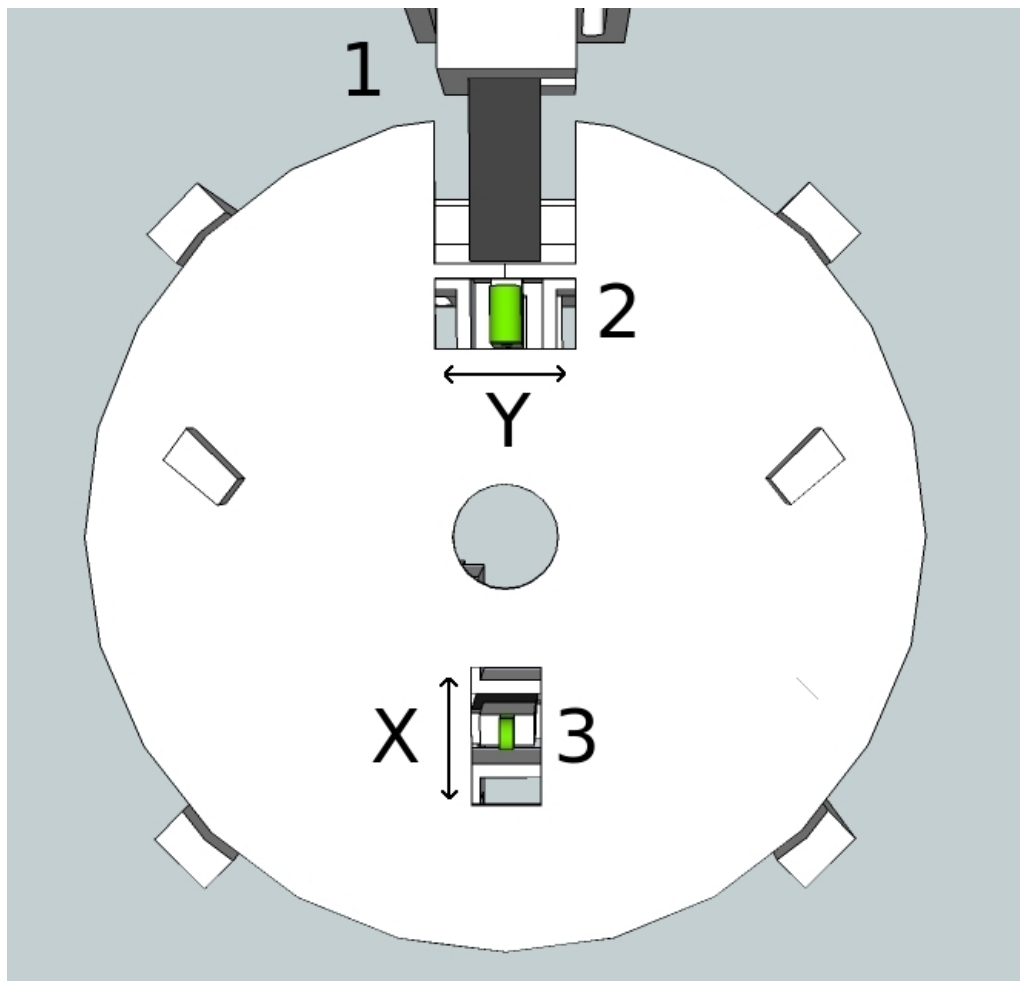


Figure 4.30: Three bender table, top view without the moving platform.

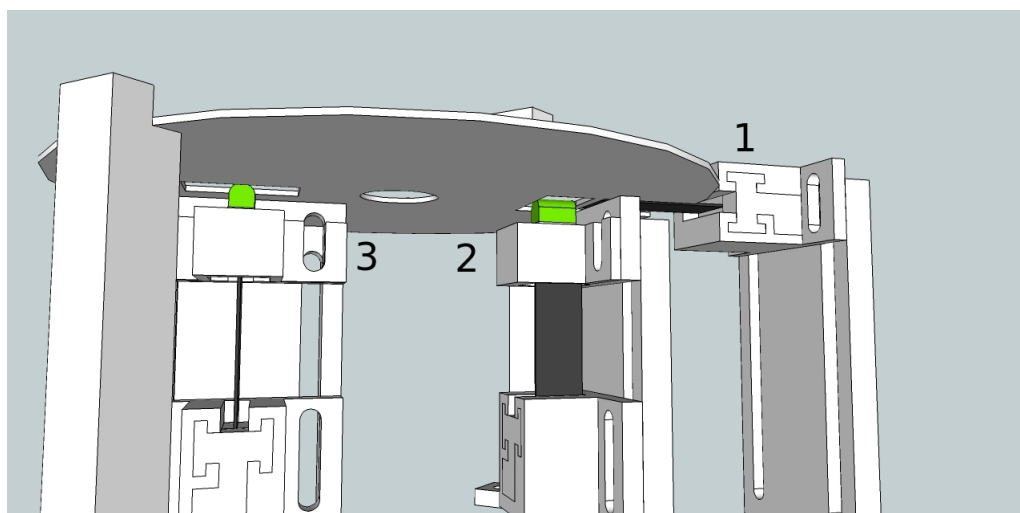


Figure 4.31: Three bender table, side view.

Figure 4.32 presents this information in a visual way. The maximum voltage (V) used can be adjusted for increased resolution (less voltage, smaller step).

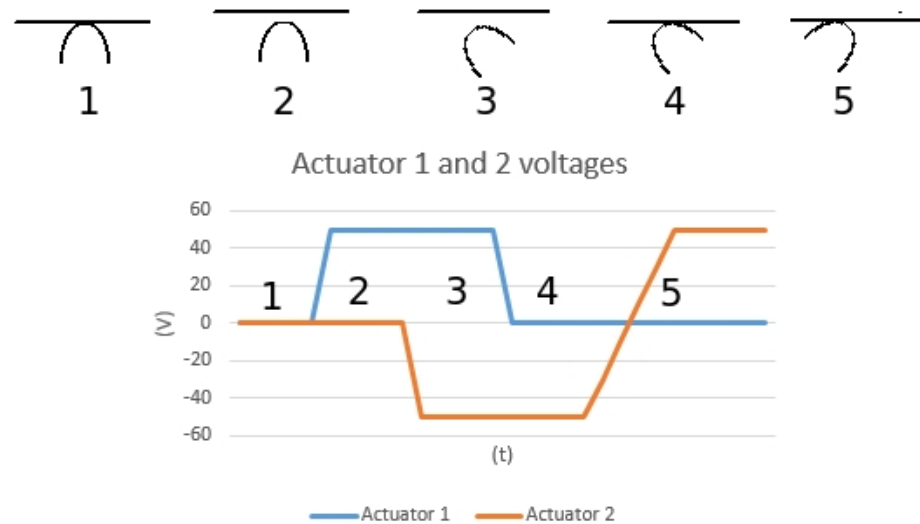


Figure 4.32: Three bender table y axis principle.

For the X axis movement, the principle is almost the same. The only difference is that the friction originated by the actuator 2 is used to the slip phase, making the actuator 1 work in reverse. This friction counters the stick movement of the actuator 3 making it slip. The procedure is:

1. The actuators numbered with 1 and 3 begin at rest (0V);
2. Actuator 3 goes to the opposite side of the movement ($+/- V$, friction from 2, platform stays in place);
3. Actuator 1 goes up ($+V$);
4. Actuator 3 goes to the side of the desired movement using the wave show in the point II in figure 4.1 (maximum value is the contrary of the one presented in 3);
5. Actuator 1 goes down (0V);

The distance between the actuators should be as big as possible while keeping center of mass of the moving platform somewhere in the middle. This will help reducing the angle of steps taken approaching the movement to a linear one instead of circular.

The full model can be observed in figure 4.34.

The platform idea is a bit raw. A holding system must be developed because, like that, the platform will be loose and subject to accidents. This will be included in the future work (see chapter 6) because there was no time to develop it.

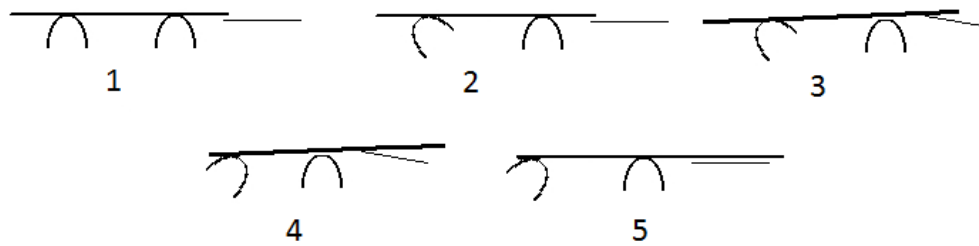


Figure 4.33: Three bender table x axis principle.

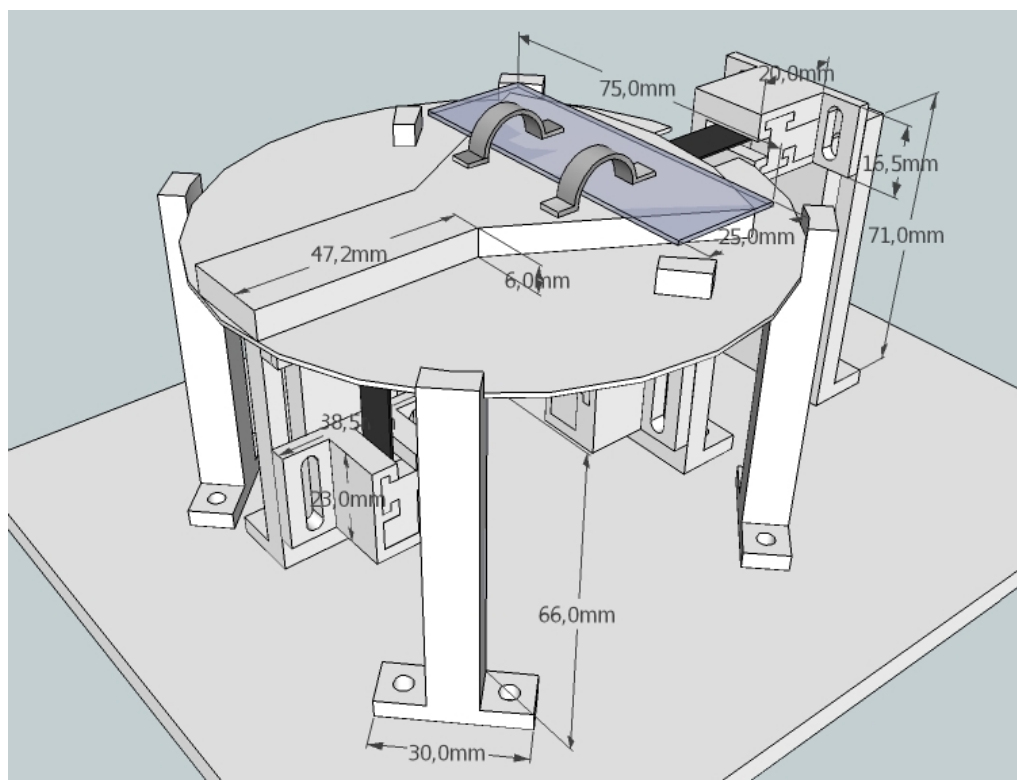


Figure 4.34: Google Sketchup three bender table.

This system doesn't rely much on the printed pieces precision and just needs manual tuning and adjustment. This was a main concern in the implementation and an ingenious approach was taken. This might not be an ideal system but is simple and easy to implement. It doesn't use any expensive material and is, almost, all printed in the laboratory 3D printer.

To conclude, for the grid movement, it changes a bit from the presented in the DC topology. The platform needs to be placed correctly before the operation of the table because there are no sensors to signalize the origin (a good idea for future work). As said before, this is a $X\theta$ arrangement so the table will do some circumference arcs. The proposed grid movement can be observed in figure 4.35. It is assumed that the user places the platform like shown in figure 4.34.

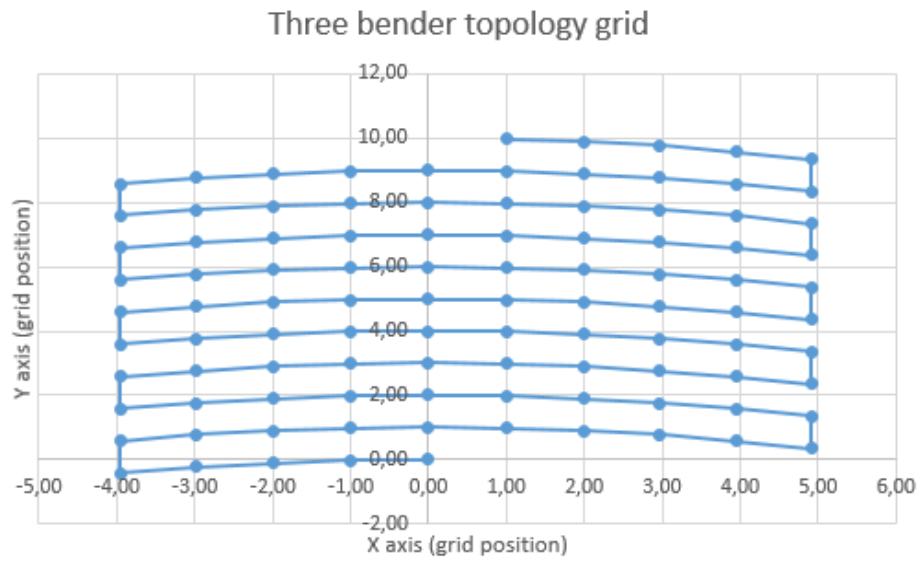


Figure 4.35: Grid movement implementation for the Three bender topology. The values represent a cell position and can be compared with figure 4.21.

4.6.2 Physical Implementation

All the pieces were printed with the same settings used in subsection 4.3.2 and the assembled system can be observed in figure 4.36.

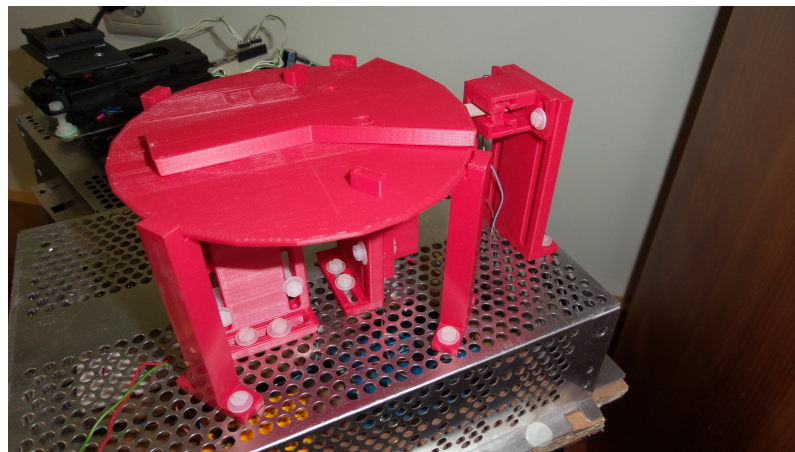


Figure 4.36: Assembled three bender table.

Like previous times, the pieces didn't fit together without some sanding. This was expected and there were no improvements in resolution. But, since this system relies in manual adjustments, it is not very critical.

The tip was changed for a full piece of rubber. This allowed easier testing without having to

print several different tip supports. While cutting the rubber piece, the 2:1 relation was always in mind. It is a lot harder to keep the proportions this way but the advantages outweigh this bit of work. Sand paper was used to give the rounded shape to the tip. After the rounding, several imperfections were added to allow more friction. A tip can be observed in figure 4.37.

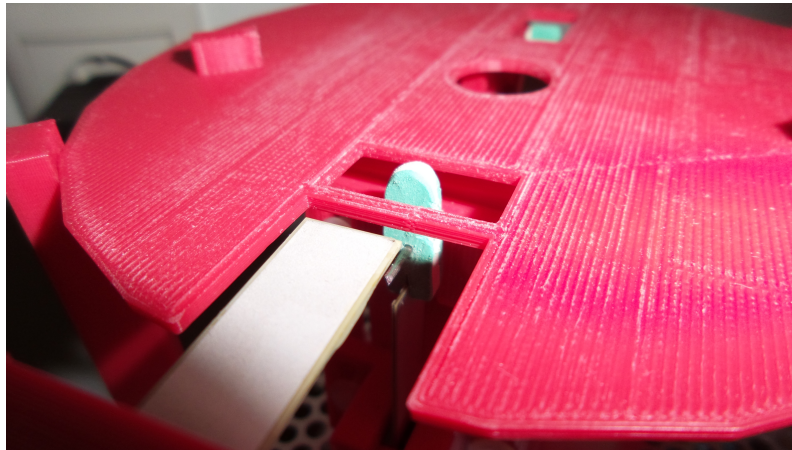


Figure 4.37: Detail of one of the tip.

To provide the friction differential between the axis, a bit of Tesa painter tape (has a rough surface) was glued to the bottom of the moving platform in the Y axis working area. As an unwanted effect, this will create parasitic friction between the support and the moving platform because they overlap each other. In a revision of the system, this could be countered by making the hole of the axis bigger (double the work area for example) removing that overlap. This idea of a bigger hole can even be used to design a holding system. If the hole has the shape of the movement, a cylinder like form can be added to the moving platform, keeping it inside the hole.

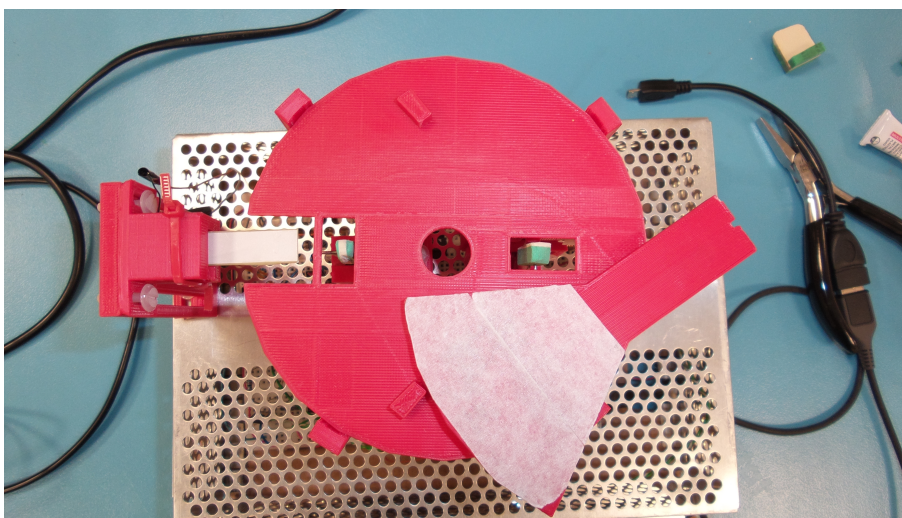


Figure 4.38: Detail of the small additions to the design.

4.6.3 Experimental Results

Like the previous implementation, after everything assembled, it was time to test the design. The smartphone was used to control everything using the application developed (see chapter 5). The step movement was coded like presented in section 4.6.

The first thing observed when testing was that the complimentary bender must be very well adjusted. This is a challenge because if it slides a bit, it loses the contact with the table and the platform will move side to side. One thing to take into account is that the complimentary bender takes its time to discharge. This is a direct consequence of the complimentary circuit used (see figure 4.29) and a $500\text{ K}\Omega$ resistor was added to improve that. If an half bridge was used, the bender could be short-circuit with the ground and a better behaviour achieved. Since there was no time left, the circuit stayed like this. In the code was added a sleep to wait for the discharge. When the complimentary bender was well adjusted and the sleep coded, the Y movement worked wonderfully. The steps were small but visible and the platform worked as expected.

As for the X axis, the results were not as satisfactory. The platform moved indeed (as will be shown) but the steps were almost unnoticeable (a lot of them were needed and, since the steps are fairly slow, this is not ideal, would take too much time). To counter this, another tips were made and tried but without any significant success. While testing, there was a detail that was obvious, the platform had its center of mass too far from the back axis. Changing it (by putting something over the platform) showed that this was indeed one of the problems. The platform moved a bit from side to side but the principle was not working (because the center of mass changed). This only showed that the platform should be very well designed or the distance between the axis actuators changed. One solution could be increasing the X axis step speed but that requires a refactor in the driver (not possible in the timeframe).

Another solution could be simplifying the X movement, adding a second complimentary bender but that would make the system even more expensive. Another option could be the use of a new system to elevate the platform (but what?). A lot of options but there was no time to pursue them.

Even with the table partially working, a grid test was made. It used the mouse process described in subsection 4.5.2.1 to measure the steps. The system assembled can be observed in figure 4.39. The black cardboard was used to allow the measurement (couldn't find a way to place the mouse over the platform) but we must take into account that, since it is assembled like a lever, a small amplification was inserted (same angle but longer radius) but, since the objective is only to get an idea of the displacement, it is not a big problem.

Run was made with 50V and with a step interval of about 5s (allows everything to get to its rest state). The results can be observed in figures 4.40 and 4.41.

It was really difficult getting these values. The assembly of the measure system was a big headache because there were times when the complimentary bender skidded messing with the

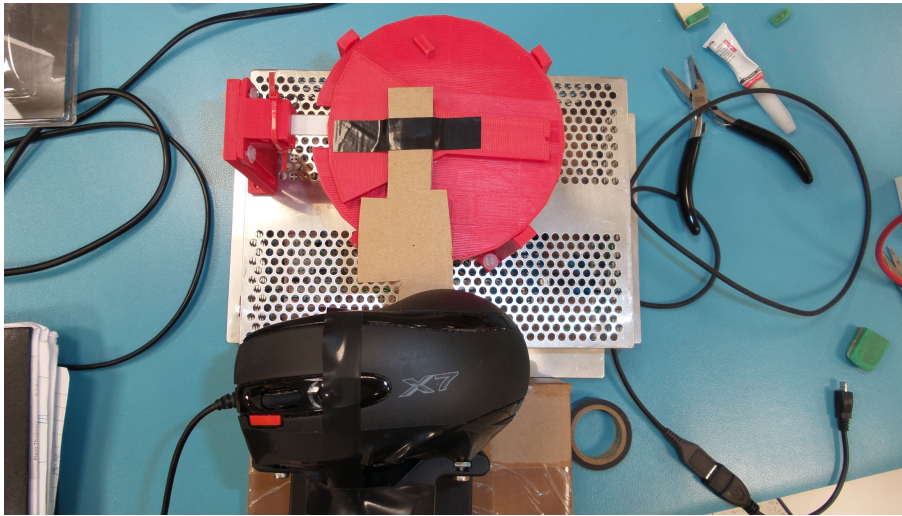


Figure 4.39: System assembled to measure the three bender topology displacement.

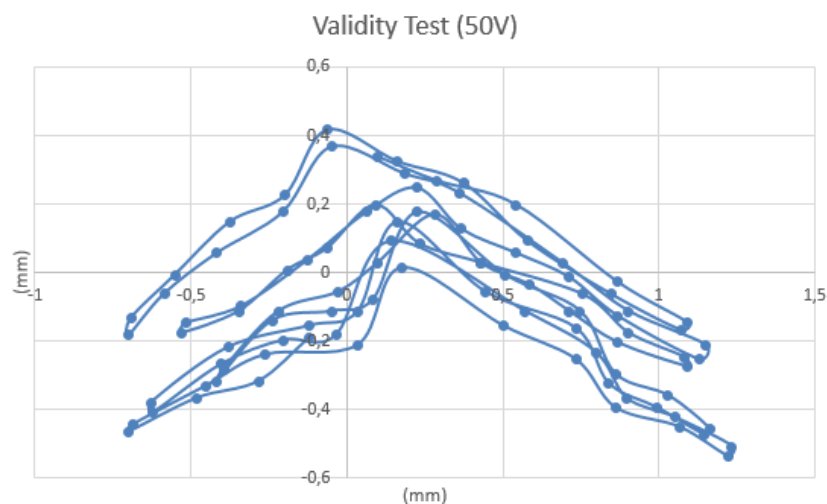
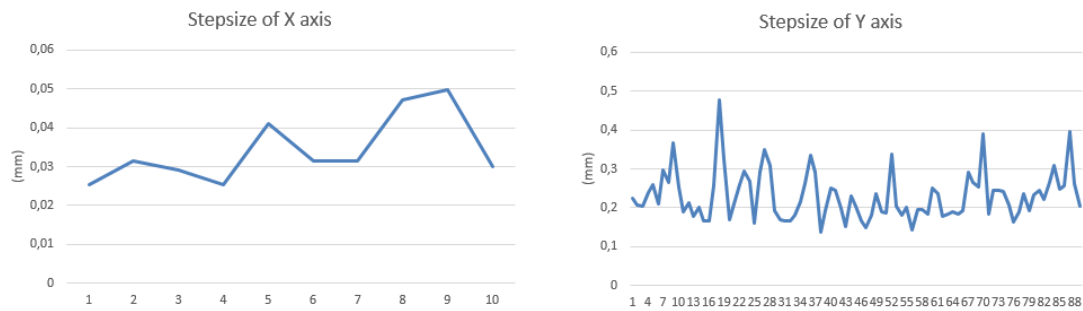


Figure 4.40: Position of the three bender table.

platform. There were a lot of problems with the cardboard arrangement too. A lot of tries were needed to get these acceptable values.

As shown, the Y movement is around of what is expected. The platform goes from side to side but without much repeatability. This could be a characteristic of the design or a side effect of the mouse or even both combined (most probable). It features steps with an average of $225 \mu m$ and that is very good (even if we add $100 \mu m$ the resolution, is enough).

The X axis movement is a mess. We can see that the platform is moving forward (the point distribution is shifting up) but that could even be a measure error of the mouse since the step is so small. The X axis measures should be ignored, they are only here as a curiosity because the mouse can't measure steps of $30 \mu m$ reliably.



(a) Step size of X axis. The average is 0.034 mm. (b) Step size of Y axis. The average is 0.225 mm.

Figure 4.41: Three bender step sizes.

It cannot be said with 100% certainty that the system has a Y axis resolution lower than $300\mu m$ but everything points to that. For the X axis, more work is needed until it works reliably. It is in the right track though and, with more time, a fully working system can be achieved and with great resolution.

4.6.4 Remarks

Summing up the three bender topology, some remarks can be made:

- The system just needs the driver and the benders. The rest is printed in the 3D printer;
- The Y axis worked wonderfully. It featured steps of under $300\mu m$ but is too dependent of the complimentary bender adjustment;
- The X axis still needs work;
- The system is not fully functional but it is in good shape for the future work. The next steps should pursue a more successful implementation of the X axis and in building a more robust system (maybe join everything in single fitting piece). This implementation showed too that the idea is valid and possible;
- During the development, stayed the feeling that the actuator was too big. Maybe a smaller bender with lower displacement but higher force is a good option for the future.

4.7 Conclusion

After all the topologies presented and discussed, is now time to discuss what is the best design, the table with benders or with DC motors.

A comparison can be made between the developed topologies:

- Both topologies were just proofs of concept, none have final designs;

- The DC motor topology was fully functional. The three bender option was almost fully functional;
- The DC motor system will need some kind of mechanical component to translate the circular to a linear movement (right now is using gears). The bender topology just uses common available materials and it's almost fully printed in the 3D printer. Because of this and without counting the actuators and drivers, the three bender topology is cheaper;
- Right now, the three bender topology needs manual adjustments while the DC motor system works every time;
- The DC motor topology can be well reduced in terms of size while the bender one is always stuck with the size of the actuators;
- The bender system features smaller steps (according to the results for the Y axis);

This was a hard fight and there is no clear winner. The bender system is a novel one and has the potential to be something really good in terms of resolution and repeatability but needs some work until a final design is achieved, while the DC motor topology is a bit more rudimentary but can make the work with a lot less effort needed. Since the final system needs to be replicable, the easy development takes a real weight in the path to take. The idea to take is that both systems are valid and represent solutions for the proposed problem.

Refer to chapter 6 for a more complete conclusion where everything is analysed and the best decisions for the project next steps are presented.

Now there is only missing the analysis of the developed software. It is presented in next chapter.

Chapter 5

Smartphone Controller Library

This chapter is focused in the Android and Arduíno communications. Before digging deep in any of the platforms code, all the assumptions made before starting the development are presented and the technologies used are shown. All the necessary functions that need to be guaranteed are presented and the communication protocol is described as well.

Then an UML analysis of the code developed for the Android is made and a high level explanation is presented for the most important functions and procedures. Every bit of code is well documented in source, this chapter is a complement to it.

After everything about Android is crystal clear, the Arduíno code is dissected and presented. Here, no driver control tasks are presented (like voltage control), they are featured in chapter 3.

Since there are two implementations, some functions and procedures are exclusive for each implementation. This is well explained below.

5.1 Technologies Used

To develop the two parts of software needed for the project, different technologies were used because both platforms have their own needs and programming languages. This is presented below.

5.1.1 Android Technologies

For the Android part, the language used was JAVA for Android. This is the main programming language for this ecosystem and brings the plus that, with little changes (only Android specific functions), the library code can be used for other platforms like Windows, Linux or MAC.

The IDE used was Eclipse with ADT plugin running under Windows or Linux (depending of the computer used for the development). This setup is the most used and well tested by Android developers. Eclipse is a well mature IDE for a lot of languages but the plugin brings a lot of new functionalities to it like a drag-n-drop GUI editor, debug tools and custom XML editors. A screenshot of the IDE can be observed in figure 5.1.

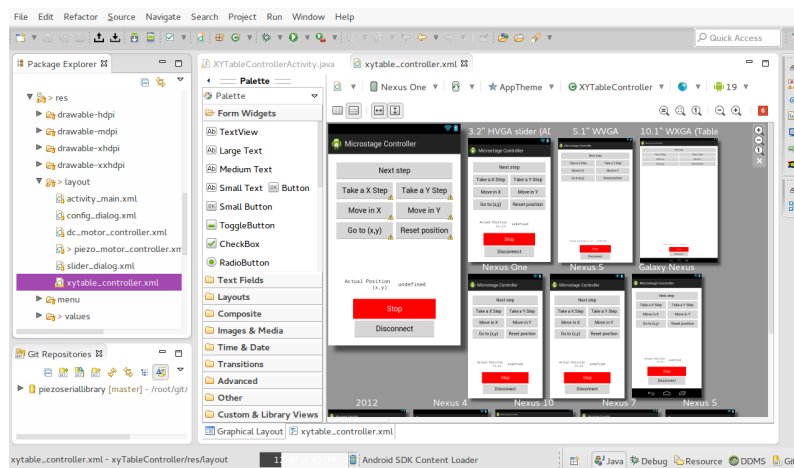


Figure 5.1: Eclipse with ADT plugin.

As referred before, the smartphone used to test the Android program was the Samsung Galaxy S4 Zoom. A picture of it can be seen in figure 5.2.



Figure 5.2: Samsung Galaxy S4 Zoom. Foto from GSMarena.com.

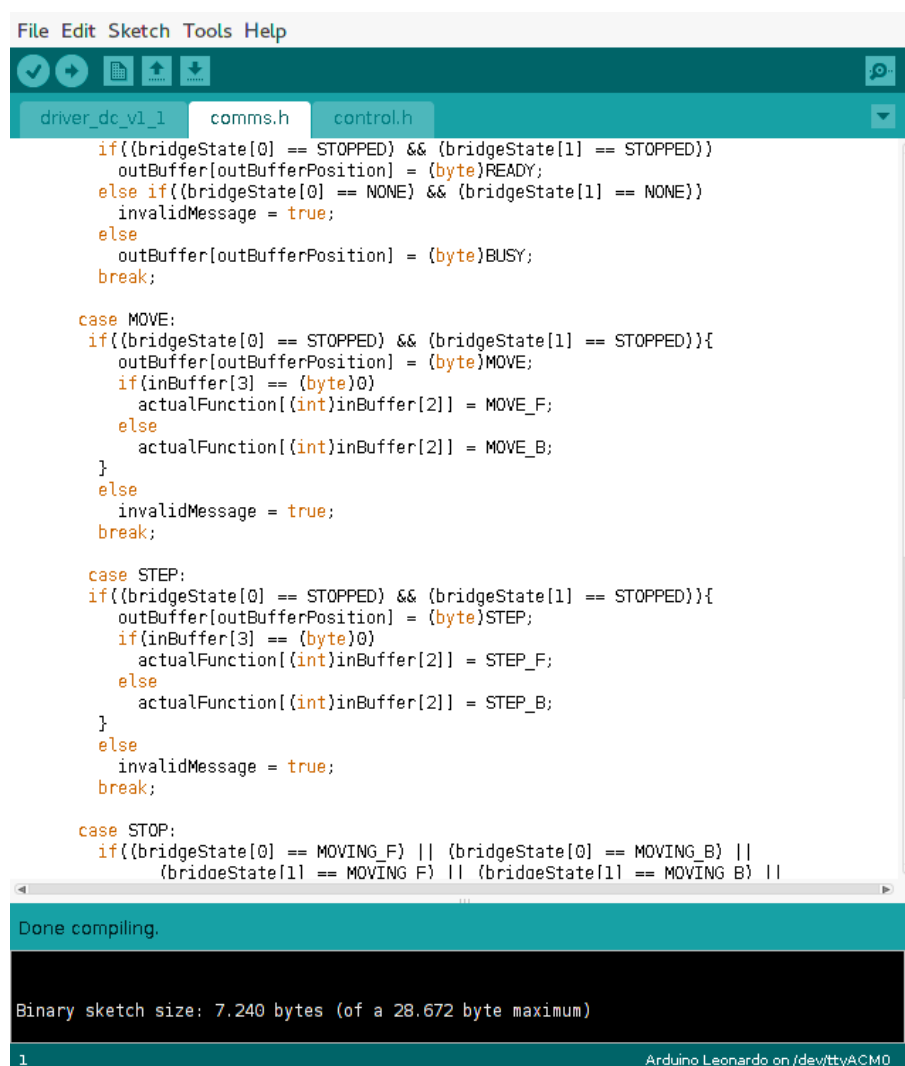
To perform the communication between both devices, the library *USB Serial for Android* presented in 2.6 was used. This library already has the raw functions like `open()`, `read()`, `write()` and `close()` so the whole protocol was built over it.

To save the work made and perform version control, a Git (distributed revision control system) repository was used. The project is private and stored in Bitbuckets.

5.1.2 Ardu no Technologies

For the Ardu no, the language used is a variant of C/C++ named Arduino Language that encapsulates most of the hard part of the microcontroller programming. Essentially, it is just a set of libraries written in C and C++.

The IDE used was the Arduino IDE. This IDE does all the basic functions needed to program an Ardu no and features a ton of libraries and a serial monitor. A screenshot of the IDE can be observed in figure 5.3.



```

File Edit Sketch Tools Help
driver_dc_v1_1 comms.h control.h
if((bridgeState[0] == STOPPED) && (bridgeState[1] == STOPPED))
  outBuffer[outBufferPosition] = (byte)READY;
else if((bridgeState[0] == NONE) && (bridgeState[1] == NONE))
  invalidMessage = true;
else
  outBuffer[outBufferPosition] = (byte)BUSY;
  break;

case MOVE:
if((bridgeState[0] == STOPPED) && (bridgeState[1] == STOPPED)){
  outBuffer[outBufferPosition] = (byte)MOVE;
  if(inBuffer[3] == (byte)0)
    actualFunction[(int)inBuffer[2]] = MOVE_F;
  else
    actualFunction[(int)inBuffer[2]] = MOVE_B;
}
else
  invalidMessage = true;
  break;

case STEP:
if((bridgeState[0] == STOPPED) && (bridgeState[1] == STOPPED)){
  outBuffer[outBufferPosition] = (byte)STEP;
  if(inBuffer[3] == (byte)0)
    actualFunction[(int)inBuffer[2]] = STEP_F;
  else
    actualFunction[(int)inBuffer[2]] = STEP_B;
}
else
  invalidMessage = true;
  break;

case STOP:
if((bridgeState[0] == MOVING_F) || (bridgeState[0] == MOVING_B) ||
  (bridgeState[1] == MOVING_F) || (bridgeState[1] == MOVING_B) ||

Done compiling.

Binary sketch size: 7.240 bytes (of a 28.672 byte maximum)

1 Arduino Leonardo on /dev/ttyACM0

```

Figure 5.3: Arduino IDE.

As referred before, the board used was an Ardu no Leonardo. A picture of it can be seen in figure 5.4.

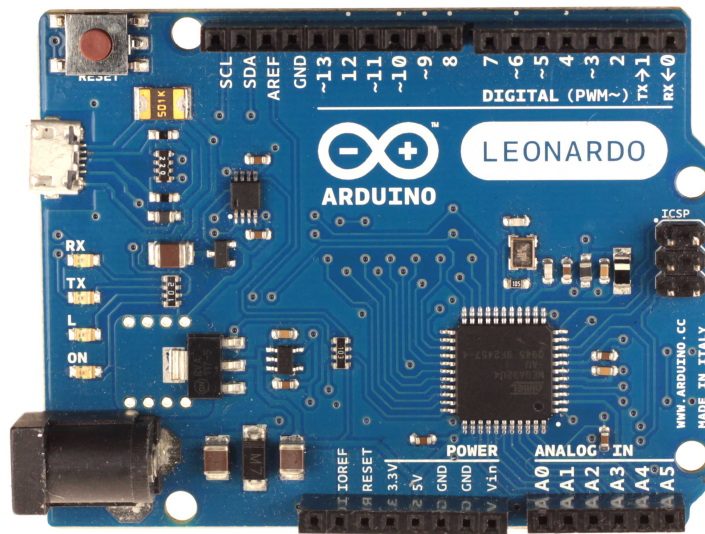


Figure 5.4: Arduino Leonardo. Foto from Arduino.cc.

5.2 Mandatory Functions

The library must implement several functionalities to allow the correct control of the two implementations (three bender table and DC motor table). Some of them are pretty standard and are related to common communication tasks (open/close communication for example). The others are specific to each topology. The common tasks that must be implemented are:

- `open(serial parameters)` - starts the communication between the Arduino and the smartphone with the supplied parameters in terms of baudrate, parity, stopbits and databits;
- `close()` - safely closes the communication;
- `isBusy()` - asks the Arduino if it is ready for a new request;
- `stop()` - stops the driver and returns the actuator to its rest position;
- `restart()` - restarts the driver logic.

The `open()` and `close()` functions are already implemented by the *USB Serial for Android* library. Just a protocol message is needed to inform the driver of the smartphone's intention.

The other functionalities are implementation based. Starting with the DC motor implementation:

- `move(axis, forward/backward)` - makes the motor move, forward or backward, in the selected axis;

- `changeSpeed(axis, speed)` - sets the PWM value in the selected axis;
- `takeStep(axis, forward/backward)` - performs a step, forward or backward, in the selected axis;
- `changeStepTime(axis, time)` - changes the step time of the selected axis;

And for the bender implementation:

- `changePosition(position)` - changes the output voltage of the driver according to the desired position;
- `goLeft(axis)` - applies the driver output voltage positively to the selected axis bender, adjusting the H-bridge (the bender goes left);
- `goRight(axis)` - applies the driver output voltage negatively to the selected axis bender, adjusting the H-bridge (the bender goes right);
- `goCenter(axis)` - applies 0V to the selected axis bender, adjusting the H-bridge (the bender goes center);
- `tableUp()` - applies the driver output voltage positively to the complementary driver (table goes up);
- `tableDown()` - applies 0V to the complementary driver (table goes down);

To understand why these functions are essential see chapter 4. Before passing to the implementation, the communication protocol is explained below.

5.3 Communications Protocol

Since this is a two device communication, the protocol was made as simple as possible, without adding any unnecessary overhead that could damage the reaction time of the system. First of all, the communication model must be defined. There are several interaction models in literature like Peer-to-Peer, Master-Slave, Client-Server or Producer-Consumer but most of them are too complex to the application. Using that idea of simplicity, a Master-Slave protocol was adopted:

- There must be a master (the smartphone) and a slave (the Arduino);
- The communication is always started by the master;
- A command is sent to the slave followed by a response from it.

This model is more than enough to power the protocol. Moving now to the protocol, as said, the master always starts the communication. It is based in very simple packets sent by both of the devices. The packets sent by the master follow the model shown in figure 5.5.

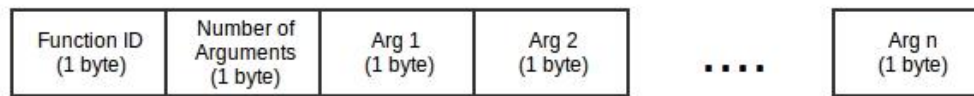


Figure 5.5: Master packets model.

As shown, this model is really simple and has everything needed to allow the communication between the two devices. The number of arguments is variable and each function named before need different ones to perform properly. This information and everything necessary to understand this protocol is presented in table 5.1.

To every request, the Arduino simply acknowledges if it was successful. For that, it sends just one byte of information containing the ID of the request sent by the master. There are other possible responses and they can be observed in table 5.2.

Rule of thumb, before a request, the master needs to ask the driver if it is ready. If this question is not made, there is the risk that the request will fail with a BUSY response. Although, some commands don't need to ask if the driver is ready. They are the OPEN, CLOSE and STOP for obvious reasons. The whole logic behind the busy messages are explained in section 5.5.

To show an example transaction, figure 5.6 presents a simple sequence diagram for a STEP request. It features every communication required to successfully request a STEP from the system.

In front of each identifier is the packet sent. Take note of the STOP request, the Arduino answers with an invalid message identifier. This is expected because the STEP request is finite in time, so the motors are already stopped when the request gets to the Arduino. But, for increased safety, that command should always be sent before CLOSE.

5.4 Android Library

After the protocol explained, here is presented the library developed for the Android ecosystem. As referred in subsection 5.1.1, the *USB Serial for Android* library is used as the backbone that powers this project. It needs a quick overview before presenting the library developed.

5.4.1 USB Serial for Android

The library is updated regularly, supports all Android devices with USB host capabilities and all serial devices that use the drivers *FT232*, *CDC/ACM* serial or *CP2102*. The Arduino Leonardo uses a CDC serial driver so its support is confirmed.

This library is built over the class `UsbManager` offered by the Android API since version 3.1. This class allows the access to the USB devices connected to the smartphone. This class originated several other ones (`UsbInterface`, `UsbEndpoint`) that allow the USB communication.

Request	ID	Number of Arguments	Arguments	Arguments Value	Explanation
OPEN	1	0	-	-	Signalizes that the system was turned on and is ready to perform.
CLOSE	2	0	-	-	Signalizes that the system is shutting down.
READY	3	0	-	-	Asks the driver if it is busy.
STOP	4	0	-	-	Signalizes that the driver should stop. In case of the DC system, the motors stop. In case of the PZT benders, the DC converter stops and the H-bridge sets every actuator with 0V
RESTART	30	0	-	-	Signalizes that the driver should restart its logic.
DC motor requests					
MOVE	20	2	motor, direction	motor - 0 if x axis or 1 if y axis; direction - 0 if forward or 1 if backward;	Commands the driver to move the selected DC motor in the selected direction. The motor only stops moving when a STOP request is issued.
SPEED	21	3	motor, speed (2 bytes)	motor - 0 if x axis or 1 if y axis; speed - PWM value, 0-1023	Changes the speed of the selected DC motor.
STEP	22	2	motor, direction	motor - 0 if x axis or 1 if y axis; direction - 0 if forward or 1 if backward;	Commands the driver to take a step with the select motor in the selected direction.
TIME	23	3	motor, time (2 bytes)	motor - 0 if x axis or 1 if y axis; time - time value in ms, 0-65535	Changes the time of the step of the selected motor.
PZT bender requests					
GOLIFT/ GORIGHT/ GOCENTER	1	motor	motor - 0 if x axis or 1 if y axis;	motor - 0 if x axis or 1 if y axis;	Applies +V/-V/0V to the PZT bender.
GOPOSITION	9	1	voltage (position)	voltage - 0 - 50	Changes the output voltage if the driver. Changing the voltage changes the position ($1V = 10\mu m$)
UP	10	0	-	-	Applies the driver output voltage positively to the complementary driver (table goes up);
DOWN	11	0	-	-	Applies 0V to the complementary driver (table goes down);

Table 5.1: Every request detailed.

Response	ID	Explanation
ID of the request	Variable	If everything is ok with the request.
INVALID	0	If the message was unknown or in an invalid time,.
READY	3	Response to the isBusy() request if the driver is not busy.
BUSY	100	Response to the isBusy() request if the driver is busy.

Table 5.2: Slave possible responses.

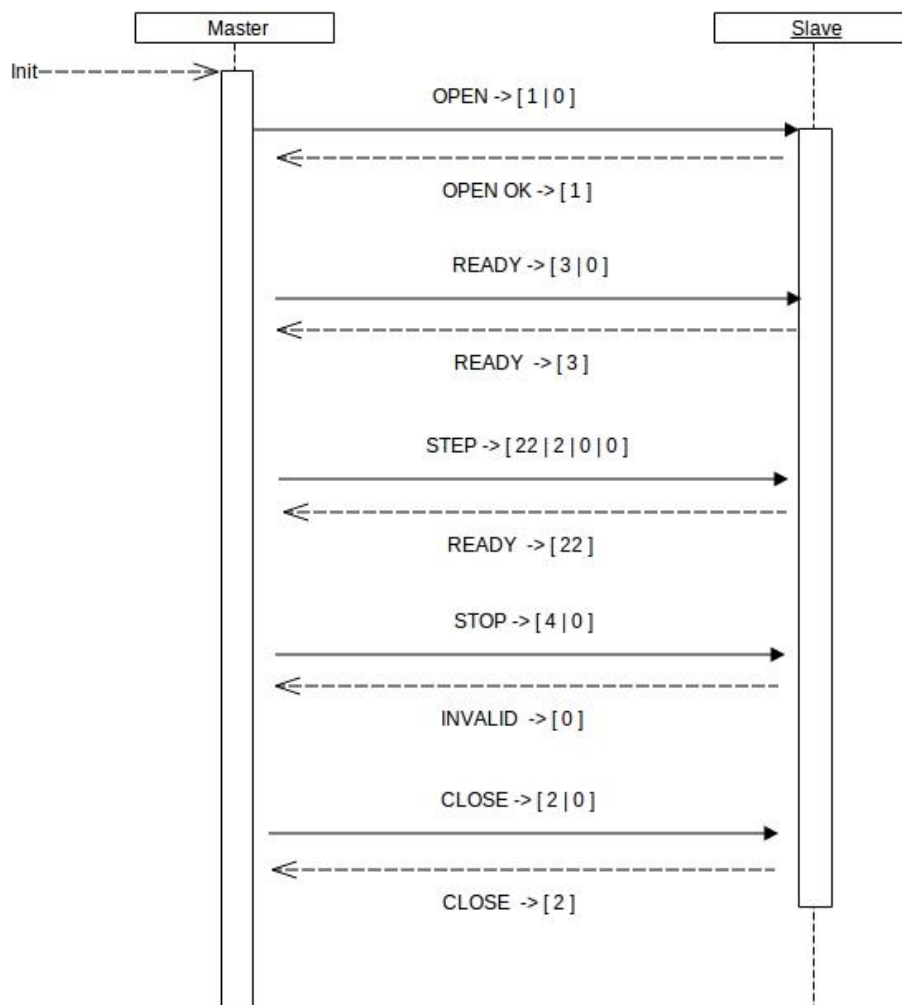


Figure 5.6: A simple request chain. This is just an example, the OPEN and CLOSE transactions just need to be made once, not at every transaction.

Using those classes, the library contains a top level interface named `UsbSerialDriver` that features all the function declarations that a driver must implement. Then a `CommonUsbSerialDriver`

class implements the shared functions from all drivers (helper functions, mainly) and each driver subclasses this top class and implement the functions `open()`, `close()`, `read()` and `write()`. Figure 5.7 represents this arrangement. Only the driver used is presented.

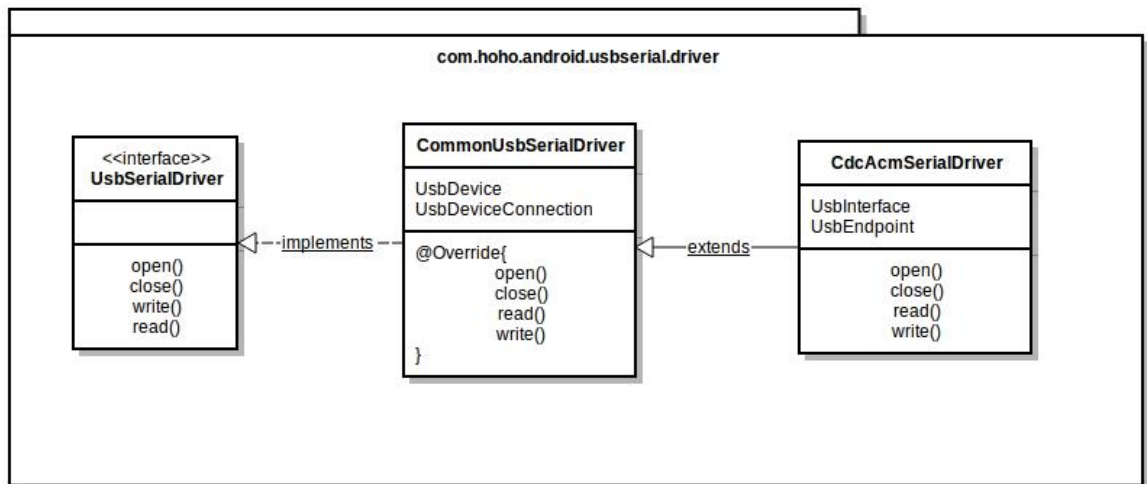


Figure 5.7: High level UML class diagram of the *USB Serial for Android* library.

The most important functions of the class `CdcAcmSerialDriver` are:

- `open()` - opens and initializes the device as a USB serial device. Without this function, nothing works;
- `setParameters(int baudRate, int dataBits, int stopBits, int parity)` - receives the parameters for the serial communication and sets them;
- `close()` - closes the serial device;
- `read(byte[] dest, int timeoutMillis)` - reads as many bytes as possible into the buffer before *timeoutMillis ms* passed;
- `write(byte[] src, int timeoutMillis)` - reads as many bytes as possible from the buffer before *timeoutMillis ms* passed;
- `setDTR(boolean value)` - very important, it should be called with *true*. Without this call, the communication isn't possible.

These functions allow the building of the project protocol. There is too another important piece that is the enum, *UsbSerialProber*. This is a helper enum that supplies procedures to enumerate the devices connected to the USB bus and gives the appropriate driver for it (none if it isn't recognized). Two functions are particularly useful:

- `UsbSerialDriver findFirstDevice(UsbManager usbManager)` - this is particularly useful because it returns the first compatible device in the USB bus. Since no more devices are connected, this will return the Arduino;
- `List<UsbSerialDriver> findAllDevices(UsbManager usbManager)` - returns all compatible devices in the USB bus. Useful for enumeration.

For more information check the [library's GitHub](#) and its Javadoc.

5.4.2 MicroStage Serial Library

Before any explanation of the implementation, the figure 5.8 presents a simplified class diagram of the library.

As can be seen in the figure, the library was created with inheritance in mind. There is a superclass named *Motor* that stores the common procedures of both type of motors. Those procedures are, mainly, interactions with the *XYTableSerialDriver* because, although the motors have different implementations of some functions, both types have to send and receive messages.

Next, there are two subclasses of *Motor* that represent each type of motor. Each one has its type of actuation and this leads to different functions. Every function does pretty much what its name indicates.

The *PiezoelectricMotor* class has a special object of the *PiezoPosition* class that tracks the position of the bender. This is made internally and the user doesn't have to worry about it.

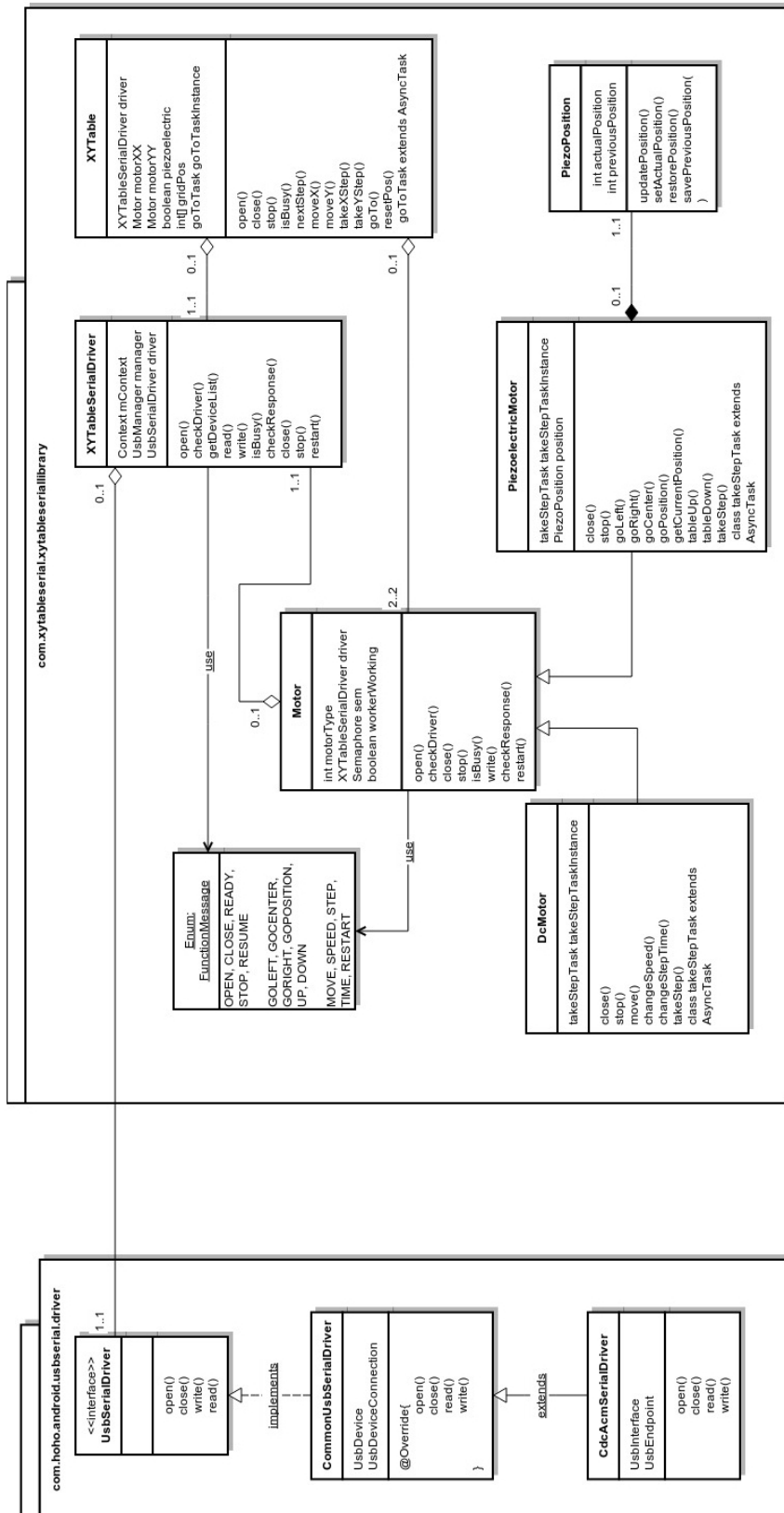
To store all the function IDs and protocol messages (see 5.3), an enum was developed. It stores all the IDs with the name presented in 5.1 and a function (`getValue()`) returns the value. Inside the enum, there are all the functions needed to create the packets and return them in a byte array.

This motor classes (*DcMotor* and *PiezoelectricMotor*, not *Motor*) can be instantiated to control just one of the axis. This was made to allow the individual debug of each motor and to whatever application the eventual user thinks of.

To tie everything up, there is the *XYTable* class. This class takes care of the table functionality, tracking the position and deciding which axis to move. The functions are straightforward and do what their name tell.

To control the communications, class *XYTableSerialDriver* is used. This class is the only one that interacts with the *USB Serial for Android* (to provide increased safety) and can't be instantiated outside of the package (it is protected). It features all the work needed to establish a connection and is responsible for the read and write transactions.

A more in-depth explanation of the principal functions and its use will be presented below. This is just an overview to the eventual user understand how the library must be used, leaving the internal details and under-the-hood implementation to a more curious reader (see its Javadoc and the fully commented source code).

Figure 5.8: High level UML class diagram of the *MicroStage Serial* library.

5.4.2.1 DcMotor class

If we want to control only one DC motor, this is the class to instantiate. It can control the X axis or the Y axis motor independently (but there is only one electronic driver board, so one motor depends of the other) and allows a custom implementation of the table control. The functions needed to fully control the motor are:

- **public DcMotor(Context mContext, int motorType)** - this is the constructor. The *Context* object is needed for the *XYTableSerialDriver* to acquire the *UsbManager*, necessary to communicate with any USB device. The *motorType* is 0 for a X axis motor and 1 for a Y axis motor. This function fails with an *InvalidMessageException* if the motor value is invalid. There is another constructor with different arguments but since one of them is an *XYTableSerialDriver* object (protected class), the user can't instantiate it;
- **public void open()** - this function is defined in the *Motor* class. Initializes the driver with the first device that finds in the USB bus (more than enough for the application), uses the default serial communication values (115200bps, 8databits, 1stopbit and no parity) and sends an OPEN message. There is a version of this function that receives all communication parameters and the device wanted (see Javadoc);
- **public void close(boolean driverToo)** - sends a CLOSE message. As said before, the message functions are protected by a semaphore. This function waits for the semaphore even if there isn't any slot available. The argument *driverToo* signalizes if the driver should be destroyed (for a user instantiating only one motor, this parameter should be true). If the *takeStepTaskInstance* is operating, stops it (prevents memory leak and safely cancels the *AsyncTask*; more about this below) calling the *stop()* function. Can fail for a lot of motives, see the Javadoc;
- **public void stop()** - sends a STOP message. Cancels *takeStepTaskInstance* if it is running. Waits for semaphore if there isn't any slot available. Most of the implementation is in the *Motor* class because it is almost the same for each motor. See Javadoc for exceptions;
- **public void isBusy()** - sends a READY message. Throws a *DeviceBusyException* if the device responds with BUSY. This function is used by almost all requests and should not be used directly because is not semaphore protected (might be changed to protected in the future);
- **public void move(boolean forward)** - sends a MOVE message. The argument *forward* signalises the direction of the movement (true for forward and false for backward). If there aren't any semaphore slots, throws a *DeviceBusyException*. Performs a driver and readiness check before sending the request and a response check after the request sent;
- **public void changeSpeed(int speedPWM)** - sends a SPEED request. The argument used in the call, *speedPWM*, must be an integer between 0 and 1023 (PWM values). If there aren't

any semaphore slots, throws a *DeviceBusyException*. Performs a driver and readiness check before sending the request and a response check after the request sent;

- **public void changeStepTime(int time)** - sends a TIME request. The argument *time* must be an integer between 0 and 65535 *ms*. If there aren't any semaphore slots, throws a *DeviceBusyException*. Performs a driver and readiness check before sending the request and a response check after the request sent;
- **public void takeStep(boolean forward)** - sends a STEP request. The argument *forward* signalises the direction of the movement (true for forward and false for backward). If there aren't any semaphore slots, throws a *DeviceBusyException*. Performs a driver and readiness check before sending the request and a response check after the request sent;
- **public void takeStep(boolean forward, int numberOfSteps)** - sends a STEP request. The argument *forward* signalises the direction of the movement (true for forward and false for backward) and *numberOfSteps* should be a positive integer for a finite number of steps or 0 for infinite steps. This function gives the user the opportunity to walk several steps with one instruction. For this, a *AsyncTask* is used, named *takeStepTask* where the *takeStep()* function is called the number of times passed to the function.

These functions are more than enough to fully control the motor.

5.4.2.2 PiezoelectricMotor class

The logic is the same from the *DcMotor* class. Almost all the functions follows the same pattern of those presented before except that they keep track of the position of the bender (using the *PiezoPosition* class). Those needed to fully control the motor are:

- **public PiezoelectricMotor(Context mContext, int motorType)** - this is the constructor. The *Context* object is needed for the *XYTableSerialDriver* to acquire the *UsbManager*, necessary to communicate with any USB device. The *motorType* is 0 for a X axis motor and 1 for a Y axis motor. This function fails with an *InvalidMessageException* if the motor value is invalid. There is another constructor with different arguments but since one of them is an *XYTableSerialDriver* object (protected class), the user can't instantiate it;
- **public void open()** - this function is defined in the *Motor* class. Initializes the driver with the first device it finds in the USB bus (more than enough for the application), uses the default serial communication values (115200 *bps*, 8 *databits*, 1 *stopbit* and no parity) and sends a OPEN message. There is a version of this function that receives all communication parameters and the device wanted (see Javadoc). This will be better explained below but the electronic driver starts with an output voltage of 50V (doesn't need *goPosition()* call);
- **public void close(boolean driverToo)** - sends a CLOSE message. As said before, the message functions are protected by a semaphore. This function waits for the semaphore

even if there isn't any slot available. The argument `driverToo` signalizes if the driver should be destroyed (for a user instantiating only one motor, this parameter should be true). If the `takeStepTaskInstance` is operating, stops it (prevents memory leak and safely cancels the `AsyncTask`; more about this below) calling the `stop()` function. Can fail for a lot of motives, see the Javadoc;

- **public void stop()** - sends a STOP message. Cancels `takeStepTaskInstance` if it is running. Waits for semaphore if there isn't any slot available. Most of the implementation is in the `Motor` class because it is almost the same for each motor. See Javadoc for exceptions;
- **public void isBusy()** - sends a READY message. Throws a `DeviceBusyException` if the device responds with BUSY. This function is used by almost all requests and should not be used directly because is not semaphore protected (might be changed to protected in the future);
- **public void goLeft()/goRight()/goCenter()** - sends a GOLEFT/GORIGHT/GOCENTER request. If there aren't any semaphore slots, throws a `DeviceBusyException`. Performs a driver and readiness check before sending the request and a response check after the request sent. This request doesn't change the output voltage it just changes the H-bridge output. The `goPosition()` function should be used for that;
- **public void goPosition(int newPosition)** - sends a GOPOSITION request. The argument `newPosition` should be an integer between -500 and $+500 \mu m$. If there aren't any semaphore slots, throws a `DeviceBusyException`. Performs a driver and readiness check before sending the request. Since the position is directly related to the voltage outputted by the electronic driver by a ratio of 10x ($100 \mu m$ is equal to 10V), there is a minimum voltage after 0V (see 3.1.2.7) that is defined in the `PiezoPosition` class and is 5V (that is converted to $50 \mu m$). This function not only sets the voltage for the driver regarding the intended position but changes the H-bridge too to match it (calling `goLeft()/goRight()/goCenter()` if needed);
- **public int getCurrentPosition()** - reads the position value from the **PiezoPosition** object and returns its value;
- **public void tableUp()/tableDown()** - sends a UP/DOWN request. If there aren't any semaphore slots, throws a `DeviceBusyException`. Performs a driver and readiness check before sending the request and a response check after the request sent;
- **public void takeStep(boolean forward, int numberOfSteps)** - executes a step request. The argument `forward` signalises the direction of the movement (true for forward and false for backward) and `numberOfSteps` should be a positive integer for a finite number of steps or 0 for infinite steps. This function gives the user the opportunity to walk several steps with one instruction. For this, a ([AsyncTask](#) is used, named `takeStepTask` were the functions above are called in the correct order to perform a step. Those functions include the

`goLeft()/goRight()/goCenter()`, `goPosition()` and `tableUp()/tableDown()` and the order can be inferred from figure 4.32 (assume everything in rest position, 0V in every bender):

- For a y step forward (assuming that the voltage is at 50V derived from a `open()` or `goPosition()` call):
 1. `tableUp()`;
 2. `goLeft()`;
 3. `tableDown()`;
 4. `goPosition()` - this function is called in a loop from `goPosition(-MAX_POSITION)` until `goPosition(MAX_POSITION)`, with `STEP_SIZE` increments. This procedure gives full control over the step size because the boundaries can be chosen at will;
 5. `tableUp()`;
 6. `goCenter()`;
 7. `tableDown()`;
- For a x step forward (assuming that the voltage is at 50V derived from a `open()` or `goPosition()` call):
 1. `goLeft()` - remember, table doesn't move because it is stuck in the y rubber tip;
 2. `tableUp()`;
 3. `goCenter()`, `goRight()` - since we can't use the `goPosition()` here because the `tableUp()` command needs the full 50V, we do two big steps that are more than capable of moving the platform;
 4. `tableDown()`;
 5. `goCenter()`;

Each instance of a *PiezoelectricMotor* controls two benders, the one controlling one of the axis and the other that lifts the table up or down. Since there is only one bender to lift up/down the table, 2 different instance of a *PiezoelectricMotor* calling this functions are controlling the same bender.

Those functions are enough to control the motor and give flexibility to perform other tasks with the benders.

5.4.2.3 XYTable class

This class encapsulates the motors in a way that the user is controlling the table and not them. Features two instances of one of the motors and it resumes to a set of simple functions that perform all the logic needed and call the correct functions from each motor.

For the XY table, there is this concept of grid. This class tracks the table in this grid that is measured by the steps taken by the table. For the *DcMotor* table, there are two sensors signaling the origin (point (0,0)). For the *PiezoelectricMotor* version, there aren't any sensors so it is

assumed that the user sets it in the proper place (point (0,0)). The grids work in different ways for both implementations. These informations are not new can be observed in sections 4.5 and 4.6.

The functions that can be used are:

- **public XYTable(Context mContext, boolean piezoelectric)** - this is the constructor. The *Context* object is needed for the *XYTableSerialDriver* to acquire the *UsbManager*, necessary to communicate with any USB device. The *piezoelectric* is true if the table is actuated by the piezoelectric benders or false if actuated by DC motors. It initializes the motors and the driver too. In case of a *PiezoelectricMotor* table, assumes that the table is at point (0,0);
- **public void open()** - this function initializes the driver with the first device it finds in the USB bus (more than enough for the application), uses the default serial communication values (115200bps, 8databits, 1stopbit and no parity) and sends a OPEN message. There is a version of this function that receives all communication parameters and the device wanted (see Javadoc);
- **public void close()** - closes both motors calling their close() function with an false argument and then closes the driver. It just needs to call the function in one of the motors because they share the same electronic driver;
- **public void stop()** - stops both motors calling their stop() function . It just needs to call the function in one of the motors because they share the same electronic driver;
- **public void isBusy()** - calls the isBusy() function in one of the motors. Throws an exception, *DeviceBusyException*, if the device responds with BUSY. It just needs to call the function in one of the motors because they share the same electronic driver. This function is used by almost all requests and should not be used directly because is not semaphore protected;
- **public void moveX(boolean forward)/moveY(boolean forward)** - makes the x/y motor move endlessly in the selected direction (if *forward* is true, it goes forward; if not, it goes backward). In case of a *DcMotor* table, calls the motor move() function. In case of a *PiezoelectricMotor*, calls takeStep(forward, 0) function. Both motors can't move at the same time, it will return a *DeviceBusyException*. Calling this function breaks the grid track;
- **public void takeXStep(boolean forward)/takeYStep(boolean forward)** - makes the x/y motor take a step in the selected direction. It calls takeStep(forward, 1) function. Both motors can't move at the same time, it will return a *DeviceBusyException*. Calling this function does not breaks the grid track;
- **public void resetPosition()** - resets the motor position. In case of a *DcMotor* implementation, calls move(false) function for each motor (it automatically stops when it gets to the sensors, it is coded in the Arduino) sequentially. For the *PiezoelectricMotor* implementation, it calls goTo(0,0) function that will be presented below. This resets the grid track broken by move() calls;

- **public void nextStep()** - according to the current position, this function calls the `takeXStep()` or `takeYStep()` in a way that the grid movements presented can be performed. When a full run is achieved, it calls `resetPosition()`;
- **public void goTo(int x, int y)** - moves the platform to a user defined point. This the arguments x and y must be valid values within the grid. Those values are defined in the filed `MAX_GRID_POSITION`. For the *DcMotor* implementation, the grid values are between 0 and `MAX_GRID_POSITION` and for the *PiezoelectricMotor* are between $\frac{(-MAX_GRID_POSITION)}{2}$ and $\frac{(MAX_GRID_POSITION)}{2}$. To implement the movements, it relies on an `AsyncTask` that calls the necessary functions in order;
- **public int[] getPosition()** - returns the current position grid values.

With this three classes, the table and the motors can be fully controlled. There are other functions in the classes but they are used for debug or aren't implemented yet. But, without a good electronic driver firmware, this library is pretty much useless. It is presented next.

5.5 Arduino Firmware

The Arduino works like a common microprocessor, runs its program in a loop like it was using a `while(1)`. The function that runs in this loop is simply called `loop()` and there goes all the code. But, before running this function, the Arduino runs another one named `setup()` where the initializations should be made. It runs only one time, at boot.

After this quick introduction, the implementation of the firmware for the table topologies need presentation. Since there are two implementations, the firmware changes a bit for each one. There was no advantage in implementing both in the same code (too ugly) so this was the path taken. As will be shown, most of the implementation follows the same ideas.

The implementation for the bender topology is based in the breadboard driver, not the PCB driver. This was made because the PCB didn't perform as expected and there was no time to debug it.

5.5.1 setup()

As said above, this function should initialize every thing needed. It is divided in two function calls:

- **setupComms()** - equal implementation for both topologies. Starts the serial communication at 115200 *bps* (can be changed at will, provided that the user changes it in the Android library) and waits that the serial port is open;
- **setupControl()** - the implementation is almost the same for both topologies, the only thing that changes are the pins. In both of them, the PWM pins have its frequency adjusted to

10 KHz thanks to the *timer* 1 (associated to pin 9) and *timer* 3 (associated with pin 5) libraries. The piezoelectric bender driver has the *ADC* frequency changed to 500KHz;

The pins follow the circuits shown in figure 3.14 and 3.25.

5.5.2 loop()

This is where all the code must be inserted. This subsection is split between both implementations because there are significant changes between them.

5.5.2.1 DC Motor

The loop section is separated in several function calls and that can be observed in figure 5.9.

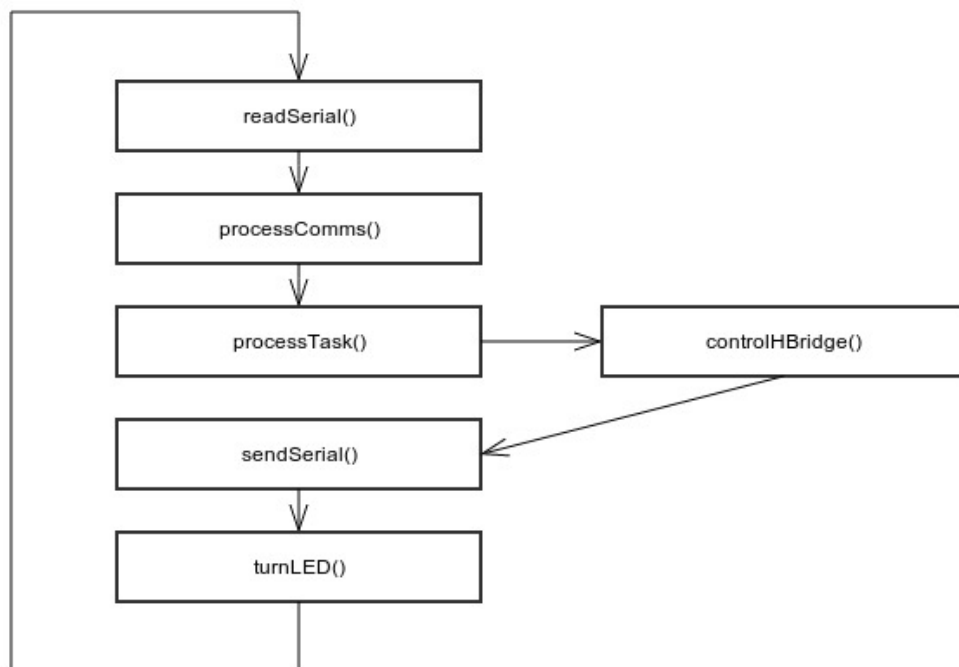


Figure 5.9: Structure of the DC motor topology loop().

1. **readSerial()** - reads the bytes (if exist) from the serial port to a byte array (*inBuffer*) and increments a variable signaling the number of bytes in the buffer. Tests if the message received is valid comparing the second byte of the message (see 5.5) and sets a boolean flag (*newValidMessage* and *invalidMessage*) to allow processing;
2. **processComms()** - processes the received message (signalized by the *newValidMessage* flag) analysing the first byte of it. If it is known and in correct order, sets two variables

(*actualFunction*[0] and *actualFunction*[1]) with the received function ID and fills an output byte buffer (*outBuffer*) with this ID (see 5.3). This *actualFunction* array controls the H-bridges, setting the transitions of a FSM that controls them. If the message is not known or sent in an incorrect order, sets a boolean flag (*invalidMessage*) that fills the output buffer with an INVALID message. If the request is a SPEED request, it calls the function *changeSpeed()* that sets the *PWM_{SPEED}* (contains the PWM speed values for each motor) array correct position with the new value. The request TIME works the same but with the *changeTime()* function and *STEP_{TIME}* array;

3. **processTask()** - just calls the *controlHBridges()* function;
4. **controlHBridges()** - this function control the two motors H-bridges by implementing a FSM (one for each bridge). It can be checked in figure 5.10. The figure explains it pretty clearly, the transitions are just the values of the array *actualFunction* (updated by the *processComms()*). The only thing that is not clear is the fact that only one motor can be moving each time. The state names are just the possible values for the array *bridgeState*. After the transitions calculation for each bridge, the *controlOutputs()* is called and there the state is evaluated and one of the *setForward()/setBackward()/setStopped()* is called. In them, the correct pins are actuated, the PWM, using the *PWM_{SPEED}*, is setted and, if the sensor is actuated, the next state set as STOPPED. See that the step logic is all performed in the Arduino firmware contrary to the PZT bender topology. This was made to reduce the jitter lag in the DC motor start and stop (not relevant to the benders);
5. **sendSerial()** - if there is anything in the *outBuffer* (signalized by *outBufferPosition* integer variable), it is sent;
6. **turnLED()** - turns the LED associated to the pin 13 (internal to the Arduino) on if both axis sensors are actuated.

5.5.2.2 Piezoelectric Motor

This follows the same structure of the last one and the functions can be observed in figure 5.11.

1. **readSerial()** - same as the DC motor implementation;
2. **processComms()** - same as the DC motor implementation but with PZT requests. If the request is a *GOPOSITION* request, it calls the function *setPosition()* that sets the output voltage to the new value, changing *expectedVC1* and *expectedVC2* variables (used in the *controlHBridges()*). If the request is a *UP/DOWN*, it calls the *up()/down()* function that controls the transistor of the complementary bender;
3. **processTask()** - calls the *controlFirstDcDc()* followed by *controlSecondDcDc()* and finishes with the call *controlHBridges()*;

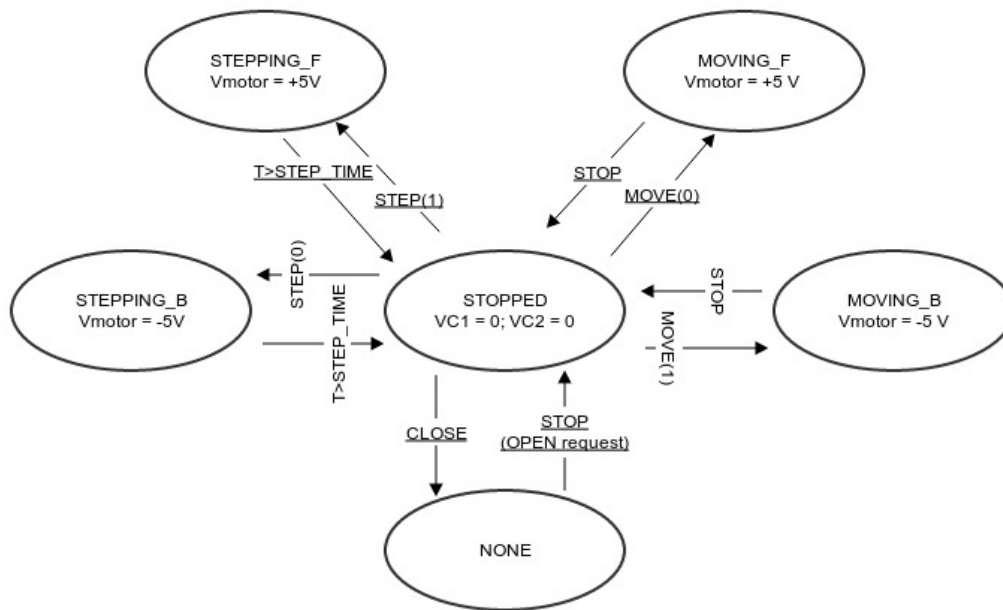


Figure 5.10: DC motor topology FSM.

- (a) **controlFirstDcDc()/controlSecondDcDc()** - reads *outputVC1/outputVC2* variable that represents the voltage that should be outputted and calculates the necessary duty cycle (using equation 3.9). Reads the real voltage at the capacitor using the ADC and, if it is higher than the wanted voltage, sets the PWM to 0% duty cycle. If it is lower, sets the duty cycle to the calculated value;
- (b) **controlHBridges()** - same as DC implementation but uses the *setLeft()/setRight()/setCenter()*. In *WAITING* state, it calls *setVoltageValues()* that update *outputVC1/outputVC2* with the received *expectedVC1/expectedVC2*. The FSM can be checked in figure 5.12;

4. **sendSerial()** - same as DC motor implementation.

This overview gives all the necessary information to understand the Arduino firmware developed. For more in-depth knowledge, see the well commented source code.

5.6 Test application

To guarantee that everything was working as desired and to provide the ability to move the table, a test Android application was developed. This application is very simple and exemplifies how the users can build their own application and work correctly with the library.

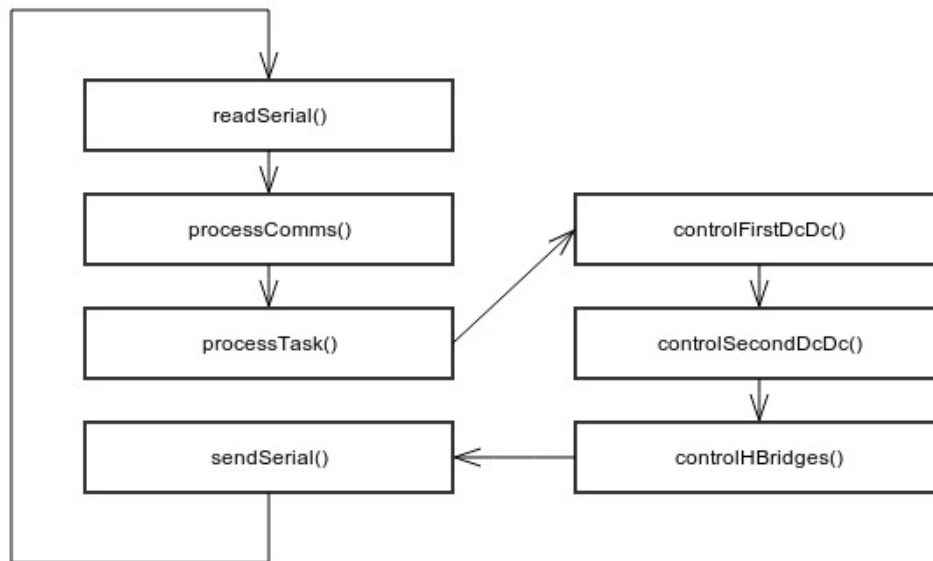


Figure 5.11: Structure of the PZT motor topology loop().

Four different activities were created to power the app. The first one can be observed in figure 5.13 and represents the welcome screen of the application.

As can be seen, it allows the choice of the device and, upon click, allows the choice of the axis wanted. At the time of the writing, there is no way to change the serial communication parameters, it uses the default ones. Choosing any of the options, the communication is established (or, if failed, the exception appears on the screen) and the correct screen appears. For the PZT option, the screen shown in figure 5.14 appears.

As shown, the screen has all the actuation functions of the *PiezoelectricMotor* class. The position control button sets the position of the bender with the help of a slider dialogue. The take step button uses a Yes/No dialogue to allow the choice of a backward or forward step. The disconnect button stops the driver, closes the connection and returns to the main screen. The white space is actually a status console where all information will appear regarding the requests and eventual exceptions. There is an indicator too of the bender position.

For the DC option, the screen shown in figure 5.15 appears.

Works the same way of the PZT motor controller and uses a slider dialogue for the time and speed requests. There is an indication signaling if the motor is moving or stopped.

Finally, the XY Table screen can be checked in figure 5.16.

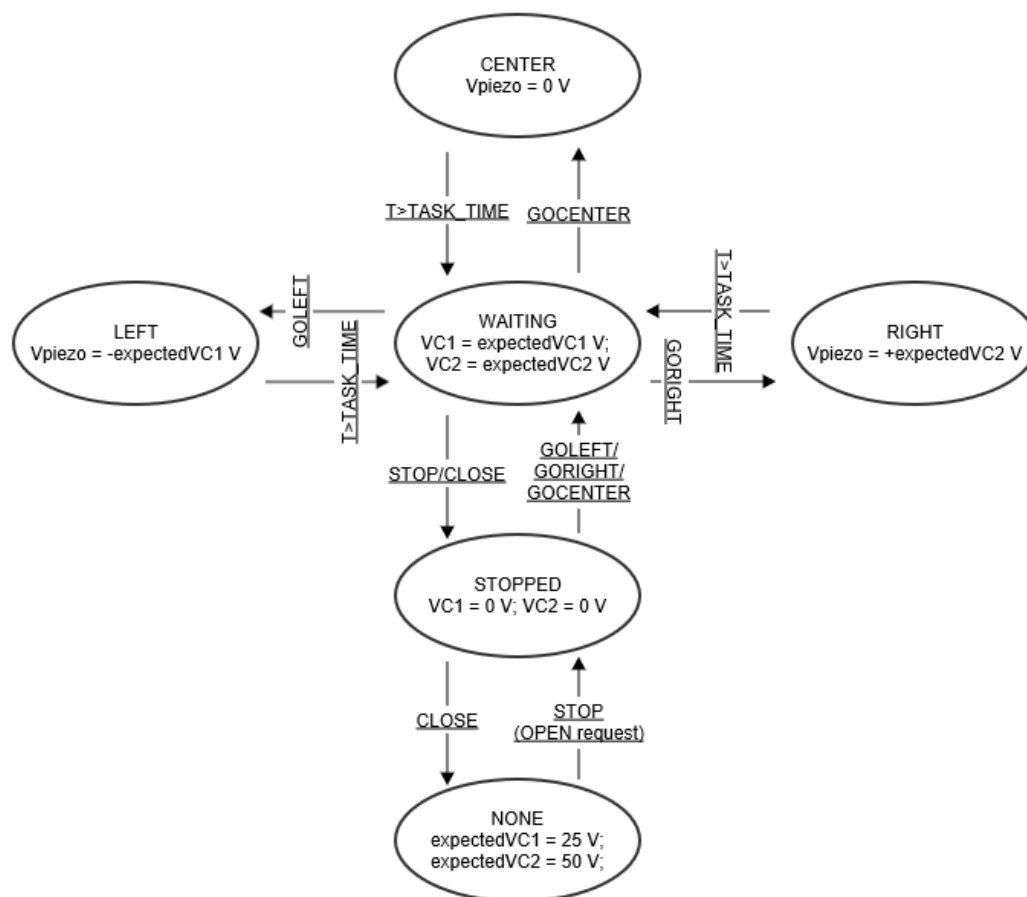


Figure 5.12: PZT motor topology FSM.

Follows the same implementation of the previous activities. The indicator shows the grid position of the table.

This is just an example of everything that can be made with the developed software. The final implementation will be very different because it will interconnect with the image processing software and not with an user.

It must be said too that this library can be easily ported to other systems (Windows/Linux/Mac) because it is JAVA based (information already presented before). Using a pure JAVA library that replaces the *USB Serial for Android* (*JavaSimpleSerialConnector* [46]), the *MicroStage Serial* works without almost any hassle.

In the ambit of this project, a PC library was started but was left at a very raw stage because the time was not enough for everything. The PZT motor controller was almost all ported and only the functions *Up()/Down()* weren't available. A screenshot of this JAVA application can be observed in 5.17.

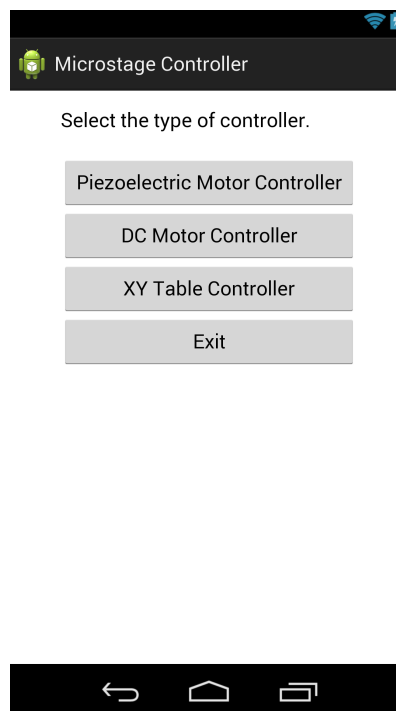


Figure 5.13: App welcome screen.



Figure 5.14: PZT motor controller screen.

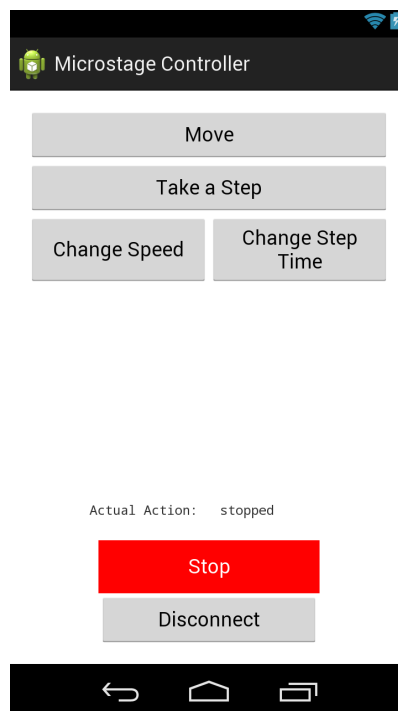


Figure 5.15: DC motor controller screen.

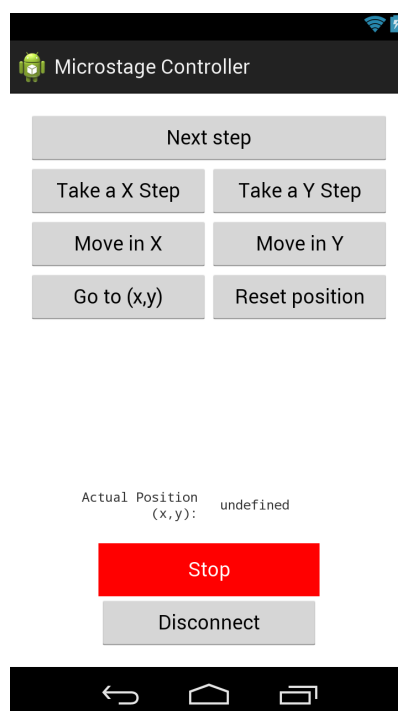


Figure 5.16: XY table controller screen.

5.7 Conclusion

After the overview of the software developed some remarks can be made:

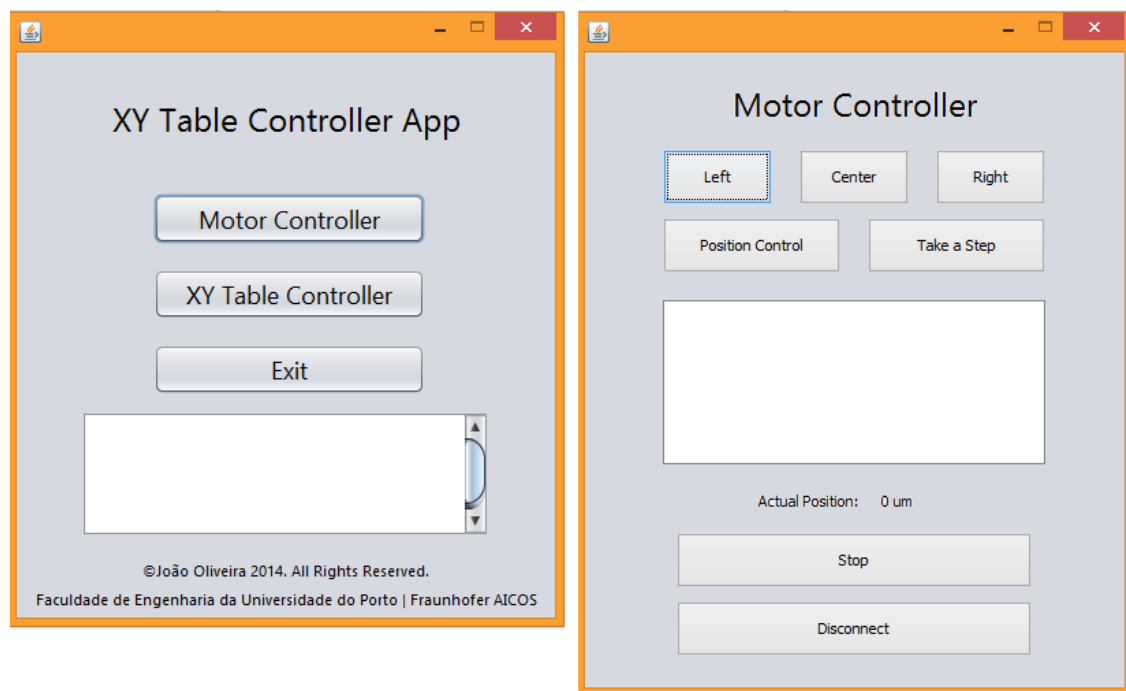


Figure 5.17: PZT motor control JAVA application running on a Windows PC.

- The developed library is enough to fully control the motors and the table;
- Some functions might be changed in the future;
- The class design is as modular as possible. It is relatively easy to add support for a new type of motor;
- It is really easy to add new functionalities to the existing classes. Just add the request in the auxiliary class (*FunctionMessages*) and code the functionality following the model from the other requests. The hard part is implementing it in the Arduino firmware (more limited);
- Some debug functions should be added in the future;
- The developed app is simple but functional. It is nowhere near perfection but does its job rather well;
- Since it is so easy to port the library for other systems, this is an added pro to it and should be done in the future.

As said before, for a more thorough examination of the implementation, the source code should be studied and the comments read.

Chapter 6

Conclusion and Future Work

This chapter is the final one where all the reflections are made about the work presented and everything is confronted with the initial objectives. Two major reflections are made with one being this confrontation and the other being the comparison between both solutions. This is really relevant because sets the tone for the presentation of the future work where some next steps for each piece of the system (electronic driver, xy table and smartphone app) are suggested.

6.1 Conclusion

After all information presented and discussed is now time to make an evaluation of the work done relating to the objectives of the thesis (remember them in section 1.3). The first and foremost idea is that not one but two possible solutions were presented, one based in common DC motors and the other in PZT benders. This already shows the success of this investigation work.

Rewinding, first a state of art review was made were some implementations of XY tables were presented and its technology studied. A very interesting comparison is made between the presented configurations and there was a focus in taking ideas for the project. In this study, a first contact with inertial PZT motors was taken and immediately jumped as a possible solution for the problem. The stepper motor configuration was a safe bet too if the PZT solution was not good enough.

Decided the actuator, an electronic driver was presented and developed. It was well sustained with a strong theoretic base and everything from the component calculation to the PCB design was made and documented. The great challenges were in developing a driver capable of supplying the necessary voltage while just using the power from the battery of the smartphone and learning how to design a PCB. The first challenge was successful and the second almost successful (refer to subsection 3.1.3). The control was easy to implement because there was already some Arduino platform knowledge.

After the driver was fully working, some time was taken in learning how to use the 3D printer. This included learning how to use a 3D modelling software (Google Sketchup), how to turn the models in printable code and how to actually print the pieces. This took a lot of time until some

satisfactory results appeared because of the perks and quirks of the Makerbot Replicator 2X. Being able to print something decent from the printer was a success by itself as can be seen in chapter 4.

Following the 3D printer headaches, some topologies were proposed, using the stick-slip phenomenon, and tested. The first one, the Triangular Tip topology, presented an interesting design (see section 4.3) where everything was expected to fit in place and with the right dimensions. This design was a possible solution but with a precise printer, not with the one available. This led to a refactor in the design and from there were born the Claw and Lever Tip topologies. Both used the same adjustable design, the difference was in the tip (see section 4.4). It used a commercial linear system because the objective was it to be a proof of concept. The Claw design was abandoned because it didn't perform and because the Lever tip was more successful. It was able to move the system but not fully apply the step movement desired. It is indeed a valid solution but needs some changes to perform reliably. At this point, two decisions were taken, build a DC motor based system and add a third bender to the PZT solution. Those decisions proven to be the best possible ones because they lead to two successful implementations of a XY table.

The DC motor system (see section 4.5) was built using old CD drives (uses the idea of being as cheap as possible) and an electronic driver was built. Everything made for this solution was a proof of concept and worked proving that the problem can be solved with DC motors. This topology achieved steps of under $300\mu m$.

The Three bender topology (see section 4.6) featured an ingenious design using a modified version of the Lever Tip and a moving platform. The use of a third bender simplified the design needs and provided an almost fully working system. With a bit more time a fully working table would be achieved but it was enough to prove that it is a successful approach and a valid solution. It featured steps smaller than the DC motor topology.

While the topologies were in heavy development, the time was divided with the Android programming (see chapter 5). It started by testing simple applications and learning the Android platform. After that, the library was started and developed. This library allows the control of both the DC and PZT motors and the XY table by communicating with the electronic driver firmware. A fully working application was presented as a way to test the developed designs.

In a general way, every objective was achieved and even surpassed. Two valid solutions were presented although their development is not finished. Both of them have their pros and cons and they are resumed in table 6.1.

As shown, both have their pros and cons but most of all and as said before, they are both responses to the problem proposed. The development even brought more paths to follow (Triangular Tip and Lever Tip topologies) and this, allied with the proposed solutions, makes it a successful research project.

As far as what is the best solution, probably the DC motor topology answers better to the challenges of the project. It is really cheap and easy to develop and more than capable of the necessary steps. This doesn't mean that the bender solution should be ditched, it is a novel proposal and worth of development. With some work, it can be improved and even surpass the DC motor solution.

Three Bender Topology		DC Motor Topology	
Pros	Cons	Pros	Cons
Novel design	Expensive driver - 17.96 €+ PCB cost	Easy	Low repeatability
Very low step size - $225\mu m$	Complimentary bender needs precision placement	Proven to work	Use of mechanical parts (gears)
High precision and repeatability	In general, hard to implement	Low step size - under $300\mu m$	Not novel
Almost everything printed in the 3D printer	PZT benders are hard to buy	Small size motors	
Use of common materials (CD, rubber)	Low autonomy time - 5h 28min	Easy to buy materials	
	Bender size	High autonomy time - 6h 51min	

Table 6.1: Proposed solutions pros and cons.

6.2 Future Work

The only thing left to do is to present some suggestions and guidelines for the future steps of the project. Every part of the system can be improved (like everything in life) and this is an important reflection to be made.

6.2.1 Electronic Driver

For the PZT bender driver, there is some work to be done. The PCB should be debugged and the problem found before any attempt on improving it. After it, the focus should go to lower the power consumption, find a way to remove the deadzone of the h-bridge (by making a bridge with individual components for example), to improve the output signal frequency and implement a more evolved control to the converter (PID for example).

Another idea that might be worth following is to explore other types of circuits in an attempt to make it consume less power. There are several common circuits that perform the task needed like camera flashes or LCD inverters that are mostly trash found in any electronics garbage. The flash circuit, for example, can provide peaks of about 3KV using a 1.5V battery. This might turn into a good solution and is worth the time.

For the DC motor, there is not much to be done, only turn it final (make a PCB shield and add some more capacitors to stabilize the current) and develop a new current limiting method (better than the LR).

A new IC should be added to provide a driver for four DC motors (or two steppers) to allow the addition of a Z axis motor.

Two market surveys should be made. One of them should address the possibility of changing the h-bridge IC to a less power hungry chip. According to the datasheet, the L293E can consume up to 60mA and that surely can be reduced. The other should address the possibility of buying an already made shield. Ardafruit has some interesting solutions (like the [Adafruit Motor/Stepper/Servo Shield for Arduino v2 Kit](#)) that need some study.

6.2.2 XY Table

Regarding the XY table, some paths are presented. The Triangular Tip topology needs a good and precise 3D printer to be fully tested. It probably can be a working solution for the project.

The Lever tip topology is a promising design and is proven to work in the literature ([42]). It needs some work in terms of the materials used for the point of contact and in the actuation signal. It might be a good idea to change the actuator to a different one and test various actuation signals.

For the DC motor topology the path is very clear. A full custom 3D printed table should be developed to remove the need of old CD-drives. A system to translate the circular movement to linear needs to be addressed too. This topology is probably the safest for a first version of a complete prototype and maybe should have more focus in the nearest future.

For the Three bender topology, some work is needed. The most important step right now is to turn the X axis into a reliable one. This will need some changes in the moving platform and the point of contact. After that solved, the design should evolve into fitting pieces, where the complimentary bender doesn't need any manual adjustments and the moving platform can be hold to the support. The focus should be divided into making the design as small as possible too (probably will need some changes in the actuator choice).

Finally, the search for innovative ways to solve the problem with PZT technology should never be stopped. At the time of the writing, there was the idea of arranging the PZT part of multiple lighters in a stack as a way to get cheap PZT actuators. This idea should be addressed to see if it is even possible.

6.2.3 Smartphone Controller Library and Application

For the smartphone part, there is only the idea of expansion. Some more functions should be added to the library (debug, statistics) but all the core ones are already there. A good idea would be porting the library to other platforms (IOS and Windows Phone). This alone would bring a lot of value to the project and increase the chances of compatible smartphones.

If the Z motor is to be added in the DC motor topology, it needs to be integrated in the library.

The integration with the vision part of the project needs to be addressed too but, since the implementation was made as transparent as possible, it will be really easy to be made.

Regarding to the image processing module, it is made in an external server, the smartphone only captures the images and sends them. Why not do everything locally? There are several

solutions that are powerful enough to join that task and even replace the Arduino as a physical controller like the well known Raspberry Pi. It features enough processing power to run the image libraries (OpenCV) and there even exists a module with a plug-and-play camera. This way, the need of a special smartphone (with good camera) is removed, and a more common one can be used as a front-end. But this goes a little beyond this project, it is just an idea that appeared during the development and needs some study.

This presentation featured some suggestions for the prosecution of the project. It is really rich and relevant because paths of improvement are shown for every bit of both solutions. But one should not settle for one of the paths, should always be looking for a chance to bring new value without any fear of redoing some of the work or trying crazy ideas. This is the core of research work and was that thought that fuelled the development effort of the thesis.

References

- [1] S. Polit and Dong Jingyan. Development of a high-bandwidth xy nanopositioning stage for high-rate micro-/nanomanufacturing. *IEEE/ASME Transactions on Mechatronics*, 16(4):724–33, 2011. URL: <http://dx.doi.org/10.1109/TMECH.2010.2052107>, doi:10.1109/TMECH.2010.2052107.
- [2] S.P. Salisbury. *Development of an XY Stage Operating in the Nanometre Regime with Millimetre Range Based on Novel Linear Piezoelectric Motors*. University of Toronto (Canada), 2007. URL: <http://books.google.pt/books?id=TjMb3jHHsMwC>.
- [3] M. Olfatnia, L. Cui, P. Chopra, and S. Awtar. Large range dual-axis micro-stage driven by electrostatic comb-drive actuators. *Journal of Micromechanics and Microengineering*, 23(10), 2013. URL: <http://www.scopus.com/inward/record.url?eid=2-s2.0-84884861782&partnerID=40&md5=c54748e421c7f661c36237480b01cf4c>.
- [4] Inc. Advanced Micro Systems. Stepper motor system basics, 2010.
- [5] Geir Andersen. Xy table, 2012. URL: <http://letsmakerobots.com/node/31381>.
- [6] Takuro Ikeda. *Fundamentals of Piezoelectricity*. Oxford Science Publications, 1990.
- [7] Physik Instrumente. Piezo actuator electrical fundamentals, 2013. URL: <http://www.physikinstrumente.com/en/products/prdetail.php?sortnr=400600.75>.
- [8] Inc. Micromechatronics. Noliac plate benders, 2010. URL: <http://www.mmech.com/noliac-actuators/plate-benders>.
- [9] Carl Blake and Bull Chris. Igbt or mosfet: Choose wisely.
- [10] Chester Simpson. Characteristics of rechargeable batteries. 2011. URL: www.ti.com/lit/an/snva533/snva533.pdf.
- [11] World Health Organization. The african regional health report: The health of the people. *The Health of the People*, 2013.
- [12] Chansuda Wongsrichanalai, Mazie J. Barcus, Sinuon Muth, Awalludin Sutamihardja, and Walther H. Wernsdorfer. A review of malaria diagnostic tools: Microscopy and rapid diagnostic test (rdt). *Med. Hyg.*, 77(6):8, 2007.
- [13] World Health Organization Geneva. *Basic malaria microscopy*. 1991.
- [14] Fraunhofer. Malariascope’s new prototype, 2013. URL: http://www.fraunhofer.pt/en/fraunhofer_portugal/news/news_archive/malaria-scope_s-new-prototype.html.

- [15] Ping Yang, Tomohiko Takamura, Satoru Takahashi, Kiyoshi Takamasu, Osamu Sato, Sonko Osawa, and Toshiyuki Takatsuji. Development of high-precision micro-coordinate measuring machine: Multi-probe measurement system for measuring yaw and straightness motion error of xy linear stage. *Precision Engineering*, 35(3):424–430, 2011. URL: <http://dx.doi.org/10.1016/j.precisioneng.2011.01.004>, doi:10.1016/j.precisioneng.2011.01.004.
- [16] Toshio Ando, Noriyuki Kodera, Daisuke Maruyama, Eisuke Takai, Kiwamu Saito, and Akitoshi Toda. A high-speed atomic force microscope for studying biological macromolecules in action. *Japanese Journal of Applied Physics*, 41(Part 1, No. 7B):4851–4856, 2002. doi:10.1143/jjap.41.4851.
- [17] Hemant Melkote and Farshad Khorrami. Closed-loop control of a base xy stage with rotational degree-of-freedom for a high-speed ultra-accurate manufacturing system. *Proceedings - IEEE International Conference on Robotics and Automation*, 3:1812–1817, 1999.
- [18] Inc. Miyachi. Xy stages, 2013. URL: <http://www.miyachiamerica.com/glossary/GlossXYStages>.
- [19] Huixiang Zhou, Brian Henson, and Andrew Bell. Linear piezo-actuator and its applications.
- [20] PiezoDrive. Piezodrive webstore, 2014. URL: <http://www.piezodriveonline.com/>.
- [21] Dynamic Structures LLC and Materials. Vf-500: Linear piezo amplifier, 2005. URL: <http://www.dynamic-structures.com>.
- [22] Thomas Soderqvist. The history of the microplate - a ubiquitous biomedical lab technology, 2010. URL: <http://www.museion.ku.dk/2010/11/the-history-of-microplate-technology/>.
- [23] Arduino SA. Arduino - homepage. URL: <http://www.arduino.cc/>.
- [24] Marek Novotny and Pekka Ronkanen. Piezoelectric actuators, 2009.
- [25] W.G. Cady. *Piezoelectricity*. McGraw-Hill, New York, 1946.
- [26] Richard Fitzpatrick. Classical electromagnetism: An intermediate level course, 2006. URL: <http://farside.ph.utexas.edu/teaching/em/lectures/lectures.html>.
- [27] J. Bert. Vibration control using piezoelectric transducers, 2007. URL: <http://large.stanford.edu/courses/2007/ph210/bert1/>.
- [28] Martin Špiller and Zdeněk Hurák. Hybrid charge control for stick-slip piezoelectric actuators. *Mechatronics*, 21(1):100–108, 2011. URL: <http://www.sciencedirect.com/science/article/pii/S0957415810001601>, doi:<http://dx.doi.org/10.1016/j.mechatronics.2010.09.002>.
- [29] Sam Robinson. Driving piezoelectric actuators. *Power Electronics Technology*, (April), 2006.
- [30] Inc. US Eurotek. Piezo actuators and electronics, 2005. URL: <http://www.useurotek.com/>.

- [31] V. V. Athani. *Stepper Motors: Fundamentals, Applications and Design*. New Age, 1997. URL: http://books.google.pt/books?id=0m8NTozFZL8C&pg=PA1&lpg=PA1&dq=stepper+motor+history&source=bl&ots=MCgWIgsLB7&sig=3HVbnifjMi2G0Pmh7GM7Qw8ok1M&hl=pt-PT&sa=X&ei=_lXpUrHqE7SM7Abbw4CgBA&ved=0CFAQ6AEwAQ#v=onepage&q&f=false.
- [32] Manuel di Cerbo. Android usb host + arduino: How to communicate without rooting your android tablet or phone, 2012. URL: <http://android.serverbox.ch/?p=549>.
- [33] mike w (Pseudonym). usb-serial-for-android, 2013. URL: <https://github.com/mik3y/usb-serial-for-android>.
- [34] Piezo Drive. 2mm range piezo bender actuator (ba4510). URL: <http://www.piezodriveonline.com/2mm-range-piezo-bender-actuator-ba4510/>.
- [35] Ned Mohan and Tore M. Undeland. *Power Electronics: Converters, Applications and Design*. Wiley, 1995.
- [36] Atmel. Avr120: Characterization and calibration of the adc on an avr.
- [37] Dan Harmon. Usb otg: The only wired interface portable consumer products need?, 2009. URL: http://www.eetimes.com/document.asp?doc_id=1272530.
- [38] Arduino SA. Arduino - arduino shields. URL: <http://arduino.cc/en/Main/ArduinoShields>.
- [39] Pooja Sharma and Amita Mahor. A new pwm based soft switching for dc/dc converters. *VSRD-IJEECE*, 2(3):6, 2012.
- [40] P. Pillay and R. Krishnan. Modeling, simulation, and analysis of permanent-magnet motor drives. ii. the brushless dc motor drive. *Industry Applications, IEEE Transactions on*, 25(2):274–279, Mar 1989.
- [41] F. Marín, F. Alhama, and J.A. Moreno. Modelling of stick-slip behaviour with different hypotheses on friction forces. *International Journal of Engineering Science*, 60(0):13 – 24, 2012.
- [42] J.Y. Peng and X.B. Chen. Modeling of piezoelectric-driven stick-slip actuators. *IEEE/ASME Transactions on Mechatronics*, 16(2):394 – 9, 2011/04/.
- [43] The Engineering ToolBox. Friction and coefficients of friction. URL: http://www.engineeringtoolbox.com/friction-coefficients-d_778.html.
- [44] Luke Chilson. The difference between abs and pla for 3d printing. URL: <http://www.protoparadigm.com/blog/2013/01/the-difference-between-abs-and-pla-for-3d-printing/>.
- [45] W.P.H Kamphuis. *Using optical mouse sensors for sheet position measurement*. PhD thesis, 2007.
- [46] Alexey Sokolov. Java simple serial connector, 2014. URL: <https://github.com/scream3r/java-simple-serial-connector>.