Final Degree Project

Biomedical Engineering Degree

Da Vinci Robot at Hospital Clinic. Haptic Devices and Performance in Robotic Tech

Barcelona, June 2021

Author: Maria Mor i Basart

Director: Dr Manel Puig i Vidal

Tutor: Dr Manel Puig i Vidal

Acknowledgments

This project would not have been possible without the involvement and help from different people, whose participation has been inestimable for the assembling and implementation of the final prototype. I would like to thank all of them.

In the first place, I do thank Dr Manel Puig Vidal, the director and tutor of this project for his involvement, patience and supervision during the whole process. Moreover, his help in proposing possible solutions has been crucial whenever a problem came up.
Secondly, I would also like to thank Júlia Meca Santamaría, a very close friend, for her company, help and teamwork with this project.

Furthermore, I do want to thank Meiling Chen as well, the EUSS student who has collaborated with us. Her help and dedication in both the 3D designing task and the final implementation of the prototype have been immeasurable.

In addition, I would also like to thank Albert Álvarez Carulla for his priceless assistance although he has never been involved in this project. His time and selfless involvement were key during the last stretch of the project development.

Last but not least, my deepest thanks to my family and friends, who have been key during these years of university but especially this last year, which has been quite atypical and hard. Thanks to their time and their unconditional support, I have been able to come this far.

"If I have seen further, it is by standing upon the shoulders of giants"

- Isaac Newton

Abstract

Minimally invasive surgery (MIS) is continuously evolving and improving its techniques due to the fact that is being more and more used given its numerous advantages, which may go from less pain and faster recovery to a better cosmetic outcome.

A big advance in this technique has been the implementation of robotically assisted systems, which have loads of advantages as well. Nonetheless, they have some perks like the lack of tactile perception and loss of depth in the vision of the surgeon. Although clinicians have adopted some means to overcome these limitations, the lack of haptic feedback is still a big problem and it's frequent rupturing healthy tissues while suturing.

Therefore, there is a huge urge and desire for the development of a robotically assisted surgical system that includes haptic feedback.

The main goal of this project is to implement haptic feedback in a robotic arm to provide the surgeon with tactile perception during the interventions and therefore prevent damages to the patient.

Table of Contents

1. Introduction

1.1. Objectives

Robotic surgery has been gaining popularity in these last decades as a way to face the existing limitations of the traditional methods in minimally invasive surgeries.

Some of the advantages that present these surgical robots are its high precision, the miniaturization of the surgery and thus, the small incisions which help to decrease blood loss, pain and the healing time.

Another advantageous characteristic of these systems is that they provide better control over the surgical instruments to the surgeon. Moreover, the software of the surgical robot may filter the surgeon's tremor.

Although these numerous advantages, some facts cast doubts on its efficiency. The lack of haptics (force and tactile) in DaVinci robotic systems is a major limitation in robotic minimally invasive surgery since it makes surgeons unable to feel the interaction between the instrument and the patient.

All in all, the main goal of this project is to implement haptic technology in the DaVinci surgical robot. The mechanical force done by the surgeon must be captured by the system and turned into a contrary force so the operator can perceive the amount of force exerted.

Another important goal of this project is to make the time of the signals' transmission fast since the surgeon needs to know to force of each movement by the time he/she is doing it.

To accomplish these objectives there are some other goals to keep in mind.

On the one hand, it's necessary to get familiarized with the da Vinci robot and the minimally invasive surgeries by attending to several interventions at Hospital Clinic. It's crucial to understand the complexity of these interventions both in the medical and technological fields to comprehend the main problems and study a solution. Doing some research about the da Vinci's evolution and the technical performances it may offer will also help to achieve the main goal of this project.

On the other hand, it's fundamental to get to know some software used in both the robotics and sensors field such as Arduino, LabView, ROS, Python and RoboDK and hardware tools, for instance, the Omni Bundle.

To sum up, the main objective of this project is:

- To develop a system to integrate haptic technology in surgical robot da Vinci.

On the other hand, the specific objectives may be resumed in the following list:

- Study of the da Vinci's surgical performance.
- Understand the problems and limitations.
- Get familiarized with software and hardware tools.
- Communicate the IMU with RoboDK to simulate surgeon's movements
- Read the torque of the servos with an external resistance
- Haptic feedback with a buzzer whose frequency varies with torque
- Haptic feedback with a vibration motor whose frequency varies with torque
- Communication between RoboDK and Arduino to establish torque and receive a haptic response at the haptic-pen.
- Communication between haptic pen and servo motors with ROS
- Validation of the final setup with UR5e

1.2. Scope

As it has been already mentioned, the main goal of this project is to implement haptic feedback to the da Vinci Surgical System. Nonetheless, it is not as easy as it may seem since, besides the technical limitations, the da Vinci System is a very complex robot.
On the one hand, it is expensive therefore, we can't take the risk of damaging it. Another thing to keep in mind is that the da Vinci System is highly restricted to any modification since there are strict clauses that state that it can't be manipulated by any other person besides an authorized technician.
Moreover, since we are talking about a surgical system, it needs to pass several approvals and certifications before being installed in an operating room. Hence, in this project, it will only be attempted the technological span since the medical it's highly restricted. Therefore, since da Vinci System can't be modified, this project will focus on implementing the haptic feedback in the UR5e robot arm, which it's allowed to get into the operating room as well.
Although the da Vinci System can't be manipulated, one can attend to several interventions to get familiar with it. Nevertheless, it has some constraints that may lead to a reduced schedule:

- The equipment is not always available.
- Due to COVID-19, not too many students may attend to the surgery. There must be one at a time.

Some other limitations that can be found in this project are time, since one year is not enough; cost, technical skills and, as previously said, medical regulations when it comes to introducing a new device.

1.3. Methodology

To carry out this project, it's going to be necessary to attend to some robot-assisted minimally invasive interventions at Hospital Clinic which is in collaboration with Universitat de Barcelona. With this, it is intended to evaluate the da Vinci System's performance and understand its limitations and how they affect the physicians.
At the same time, bibliographic research will be done to learn more about da Vinci's output, features and technical and mechanical principles.
Concurrently, there will be technical sessions at the Biomedical Engineering and Electronics Department at Universitat de Barcelona, where it will be studied the different approaches of the haptic technology implemented in a robotic arm bought by the department last year.
Although the best thing would be to attend several days per week, due to the COVID-19 the schedules of the department and the capacity may be restricted. Thus, besides the face-to-face sessions, there will be online meetings to evaluate the progress of the project.

2. State of art

2.1. History and evolution of Minimally Invasive Surgery

Minimally invasive surgery involves several techniques to reduce the size of the incision to decrease the healing time of the wound. Not only that but also the fact of reducing the patient's pain. Thus, these techniques provide numerous advantages in front of open surgery. On the one hand, it should lead to less operative trauma and complications. On the other hand, as has been already mentioned, it causes less pain and speeds the recovery.

These medical procedures include, among others, endoscopy, laparoscopy and arthroscopy.

The first endoscope with a light source was produced by Philip Bozzini in 1806. In this first approach, the system was composed of a group of mirrors which reflected the light from a candle inside an aluminium device to the point of interest.

Although it was considered the first true endoscope, it wasn't accepted by the medical community since no one noticed the big potential this device could have had. Nonetheless, Bozzini's study was crucial for establishing the background for future studies in the endoscopy's field.

In 1826, there were two main advances. On the one hand, Pierre Salomon Segalas presented a cystoscope based on Bozzini's first endoscope. On the other hand, John Fisher was developing a vaginoscopy instrument in Boston. He thought it could be better to use a vaginoscope to evaluate the cervix of a shy woman instead of the standard exposure.

In the mid-1800s, Antoine Jean Desormeaux began using a modification of the first Bozzini's endoscope for urologic procedures by changing the light source. In this case, instead of being a wax candle, it was a flame from an alcohol and turpentine solution which ended producing a brighter beam of light.

For the rest of the 19th century, several modifications of these firsts approaches of endoscopes for different examinations were produced.

It was in the early 20th century, in 1901 when the first laparoscopy intervention was performed and even though it was done with an animal, it was crucial for establishing the importance of a sterile pneumoperitoneum (insufflation) to allow visualization, which would end up being fundamental for future laparoscopies.

Not far away, the bases of minimally invasive surgery in humans were being established by Hans Christian Jabobaeus who in 1911 published a paper about laparoscopy and thoracoscopy in humans.

Nonetheless, with the advances in medicine and pharmacology, some of these procedures stopped being so popular. For example, thoracoscopy began being less used when streptomycin was discovered.

With the years, more physicians and scientists kept improving those instruments with more advanced technology and better mechanical properties. A huge advance in these devices was in the early 1960s when an electronic carbon dioxide insufflator was produced.

Nonetheless, it was in 1982 when a major advance was done: a high-resolution video camera, which sent the images to a monitor, was installed in an endoscope. This allowed an increase in the surgeon's visual field which, until then, it had been very limited since the surgeon had to hunch over and peer into the scope.

In the late 20th century it started a revolution in the minimally invasive surgery. Loads of the instruments previously mentioned suffered major changes and technical improvements. However, it's in the 1990s when there's the first robot-assisted surgery used for laparoscopies [1].

2.2. Evolution of da Vinci Surgical System

The da Vinci Surgical System is a robotic surgical system developed by Intuitive Surgical, an American company founded in 1995 to create robotic-assisted systems that help improve physicians' surgery performances by using less invasive techniques. Although there is a lack of evidence that robotic surgery leads to better long-term results following laparoscopic surgery, the da Vinci System is being used nowadays for several types of interventions: radical prostatectomy, pyeloplasty, cystectomy, nephrectomy and ureteral reimplantation; hysterectomy, myomectomy and sacrocolpopexy; hiatal hernia repair and transoral robotic surgery for head and neck cancer, among others [2].
This last procedure was approved by the Food and Drug Administration (FDA) in 2009. Nonetheless, it was approved by the FDA in 2000 for general laparoscopic surgeries.

The da Vinci System is controlled from a console by the surgeon, which helps to overcome the limitations of open laparoscopic intervention by improving the physician's vision, precision and control [3].
The da Vinci System is composed by the console, as it has been already mentioned; the patient-side cart with three to four interactive robotic arms, which contains the EndoWrist instruments and the endoscope; and the vision tower, which includes the central processing system display equipment and other surgical instruments such as electrosurgical units and insufflators [3].

Besides the high precision and accuracy this system allows, the vision of the surgeons is also improved since they can control the camera and focus it wherever they need to.
This systems include the main components of a conventional laparoscopy in order to perform small incisions, introduce the instruments through the trocars, insufflate CO2 into the abdominal cavity and use long tools. Moreover, they also provide with specific tools, which are the EndoWrist (see Figure X). This tools have 7 degrees of freedom and try to simulate the hand movements.
Besides these properties, some versions of the da Vinci system include other features such as vessel seller, fluorescence to easily identify tissues, simulators to practice, single-site option that allows only one invasive entry, etc.

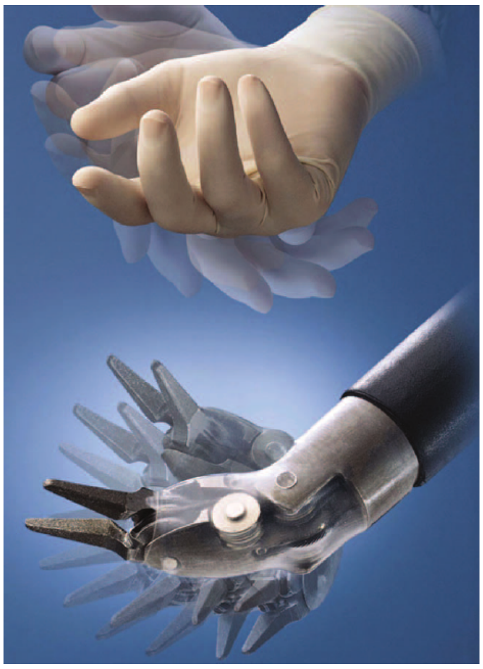


Figure 1. Comparison of the EndoWrist with surgeon's hand. Extracted from https://www.researchgate.net/figure/EndoWrist-O-instrument-7-df-just-like-the-human-wrist-a-2008_fig3_23657822

There are five versions of the da Vinci System. Nonetheless, the first one, which was the Standard, stopped being commercialized in 2007. In 2006, appeared the 'S' version, which was considered the second generation of da Vinci Surgical System and included improved features concerning its previous version: faster set-up times, rapid instrument exchange and multi-image display capabilities [4].
Later in 2011, the 'Si' version was introduced to the market. In this edition, several features were included: dual-console capability, which allowed the training and collaboration during surgeries; enhanced high-definition 3D vision with up to 10x magnification and an immersive view of the operative field; and an updated user interface [5].
In 2014 appeared the fourth version: da Vinci 'Xi'. Here, the surgeon console allows controlling the 3D endoscope and the EndoWrist instruments through controls and pedals. It also replicated perfectly the alignment of the eyes, hands and instruments. The fourth-generation also allowed for many other things, such as the elimination of the physicians' hands tremors and involuntary movements; more advanced instrumentation, vision and features like integrated table motion and multi-quadrant access [6] [7].
The latest version is da Vinci System 'X' (see Figure 1) which added new features to improve its precision and maneuverability. Its arms are thinner than its predecessors, which allows for higher mobility during the intervention; it also includes voice-guided systems and laser [8].

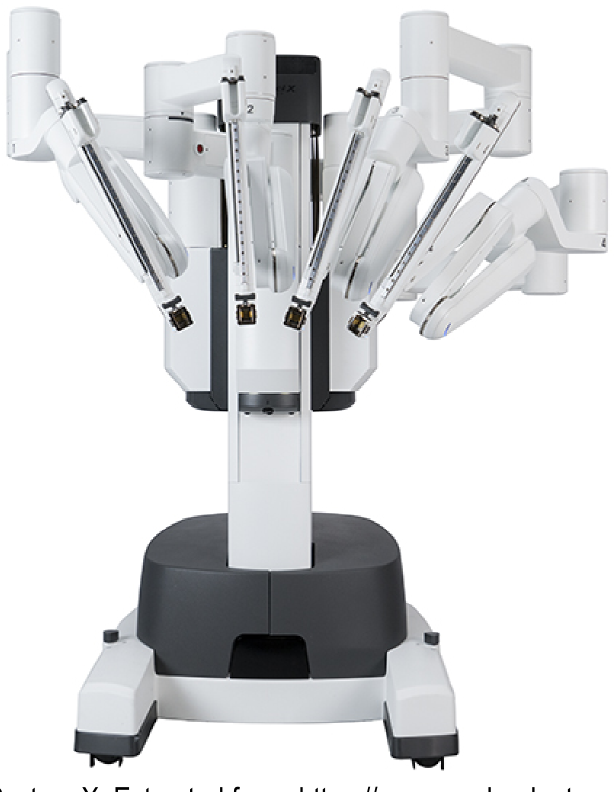


Figure 2. Da Vinci System X. Extracted from: https://www.medgadget.com/2017/05/intuitives-new-budget-friendly-da-vinci-x-robotic-surgical-system-cleared-u-s.html

As it has been previously mentioned, some versions include new features which are called “new technologies”. This entails loads of benefits in front of traditional methods, such as less pain, lower risk of infection, shorter post-surgical hospitalization, better cosmetic outcome since scars are smaller, etc.

Given these advantages, da Vinci surgical system is currently being used in several procedures [2]:

- Radical prostatectomy, pyeloplasty, cystectomy, nephrectomy and urethral reimplantation.
- Hysterectomy, myomectomy and sacrocolpopexy.
- Hiatal hernia and inguinal hernia repair.
- Gastrointestinal surgeries such as resections and cholecystectomy.
- Transoral robotic surgery for head and neck cancer.

Although it may not be as used as in the previous clinical procedures, da Vinci is also being implemented in cardiology procedures such as coronary bypass and mitral valvular reparation.

2.3. Haptic technology

As a brief introduction, it's worth to mention that in the early 20th century, haptic psychophysicists was used to describe the field that studied human touch-based manipulation and perception but it wasn't until the 70s and the 80s that robotics didn't start focusing in perception and manipulation by touch [9].

Robot-assisted minimally invasive surgery (RMIS) presents some advantages in front of conventional surgery. On the one hand, it improves the accuracy and dexterity of the surgeon. Nonetheless, it has a major limitation: the lack of haptic feedback. In RMIS the surgeon loses completely the natural haptic feedback since he/she no longer manipulates the instruments directly. Hence, the integration of this technology in robotic surgery systems is one of the main goals nowadays: the surgeon feels his/her own hands contacting the patient. For this, haptic sensors and haptic displays are necessary. The first component is in the patient-side and acquires the haptic information. On the other hand, haptic displays send the information to the physician [10].

Haptics may provide two types of information: tactile feedback, which refers to the information acquired by the sensors connected to the user's body; and kinesthetic or force feedback, related to the forces and positions of the muscles and joints. This information includes temperature, distributed pressure, force, vibrations and texture.

Regarding the force feedback, it measures the forces applied to the patient by the surgical instruments and provide a contrary force to the physician's hand through the device. Nowadays these sensors work by measuring forces and torques. However, it's difficult to add them to already existing robotic systems that weren't designed with this purpose. Nonetheless, some researchers have designed new tools that may be attached to existing instruments. Although the advantages these sensors may provide, there are still some limitations. One of them is the fact that some robots may have seven degrees of freedom, however, not all direction can provide force feedback. Nonetheless, the most crucial limitation is the equilibrium between the system stability and the transparency for force feedback since, as soon as we acquire more transparency, some errors and delays may appear. Thus, other approaches have been considered such as audio feedback, graphical feedback, vibrotactile display, etc [11]. Moreover, the surgeon can obtain more information by observing how the patient's tissue reacts to the movements with the surgical instruments.

On the other hand, we have tactile feedback. That, although it might not be as important as force feedback, it is interesting for palpation. Here, the tactile sensors can detect some mechanical

properties of the tissue such as viscosity, texture or compliance or other features like pressure distribution and the deformation of the area being manipulated.
However, as it has been already mentioned there are some limitations. Besides the lack of perfect transparency, there are some other constraints to keep in mind: cost, size, geometry, biocompatibility and sterility are the main ones. This is why haptic feedback for RMIS is still under study in engineering labs and it's not ready for clinical trials yet [10].

Although it may not be fully implemented in the healthcare industry, there are other industrial sectors where haptic feedback and tactile perception are implemented in robots.

One example would be *HaptX*, a company that has developed a glove with true-contact haptics. Their glove contains a technology that allows our skin feel the same way a real object would through its more of 133 points of tactile feedback per hand. This product may apply up to 174 N of resistive force per hand and has more that 36 DoF. As previously mentioned, it has no application in the medical field yet, but it's widely used for robotic integrations since HaptX gloves may control robotic hands, grippers and arms [12].

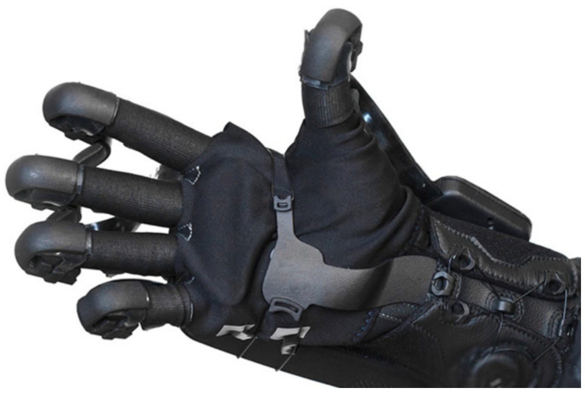


Figure 3. HaptX Gloves DK2. Extracted from: https://haptx.com/

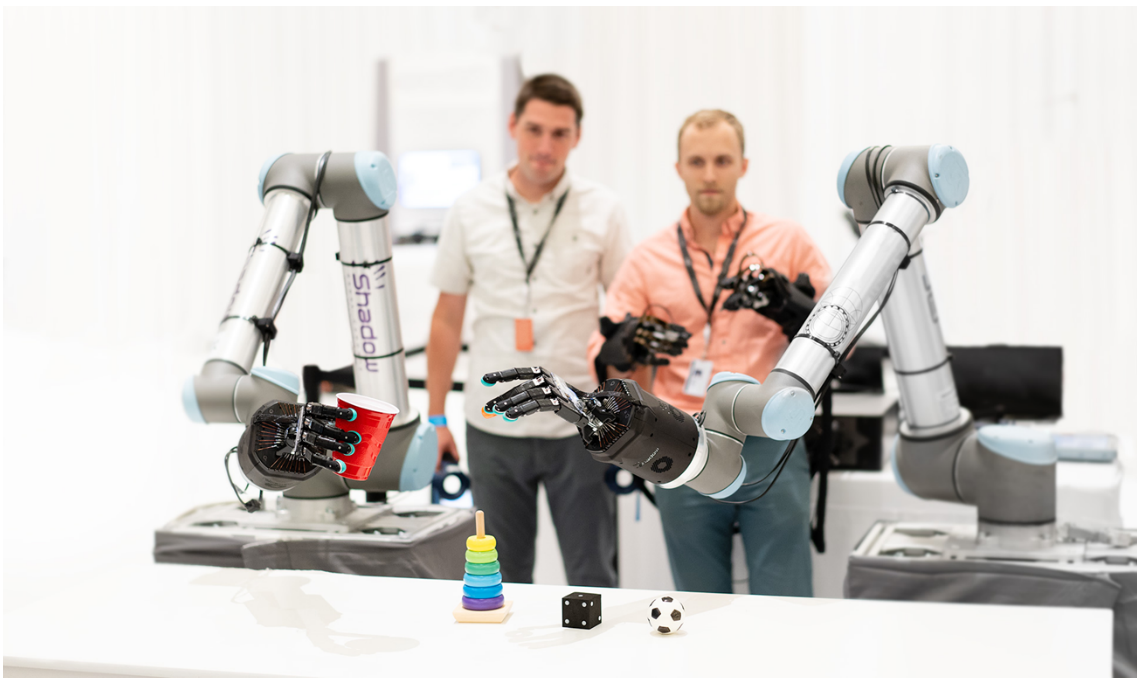


Figure 4. Application of the HaptX Gloves DK2 with a robotic arm. Extracted from: https://haptx.com/

3. Analysis of the market

3.1. Historic evolution of the market

As it has been mentioned in the previous section, robots were introduced in the minimally invasive surgery field in the 1900s. Its appearance emerged with the demand for more precise and safer operations and the willingness of the surgeons to adopt more MIS techniques to improve their results [13] [14].

Although the first surgical robot, PUMA 560, was developed in 1985 for a brain biopsy, the first robotically assisted MIS was performed for prostate's interventions in 1991 by the robotic system Probot developed by the Imperial College of London [14].

This system meant the introduction to the second generation of surgical robots and although it signified a huge advance it couldn't equalize the traditional MIS for several limitations, for example, the lack of haptic feedback [14].

During the decade, two endoscopic robots made a difference: The Zeus robotic system by Computer Motion and da Vinci robotic system by Intuitive Surgical Inc. They both became commercially available in 1998 and 2000, respectively.

The Zeus system was an improvement of AESOP, also from Computer motion, used in the first beating-heart coronary intervention in Canada and the first trans-Atlantic operation with a telerobotic system with the robotic device in New York and the patient in France. This arose the competition between both companies which eventually ended with Intuitive Surgical Inc. acquiring Computer Motion in 2003 and becoming the leader in the sector and most widespread MIS robot worldwide.

Da Vinci Surgical Systems is currently installed in 4986 units worldwide and the 'Si' version costs around US$2 million plus the costs of annual maintenance fees, which are hundreds of thousand dollars and although it's the leading player, there are some other companies operating in the global surgical robot's market such as Medtronic, Verb Surgical, Stryker, THINK Surgical, etc. Nevertheless, none of them has yet implemented haptic feedback technology. Despite this, in 2012 Rob Surgical was created in Barcelona with the aim of solving some of the main RMIS challenges. Between 2012 and 2018 they developed the Bitrack system (see Figure 2) with a sensory feedback function, which tries to improve today's robots efficiency by using new technology, improving its usability and reducing the acquisition costs. Nonetheless, they're still in the regulatory phase to obtain CE and FDA certifications. Hence, it won't be in the market until 2022 or 2023 [15].

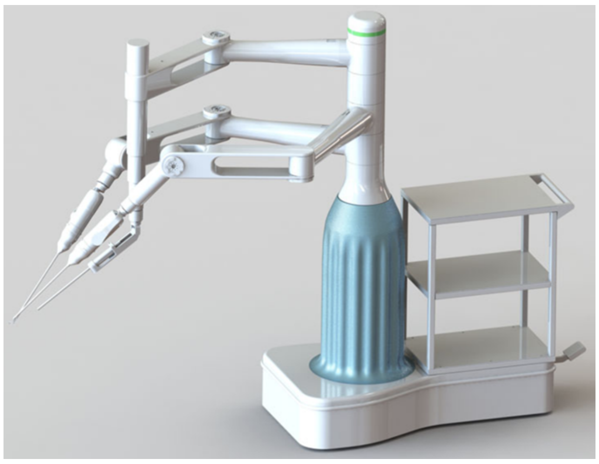


Figure 5. Bitrack from Rob Surgical. Extracted from: https://www.robsurgical.com/bitrack/

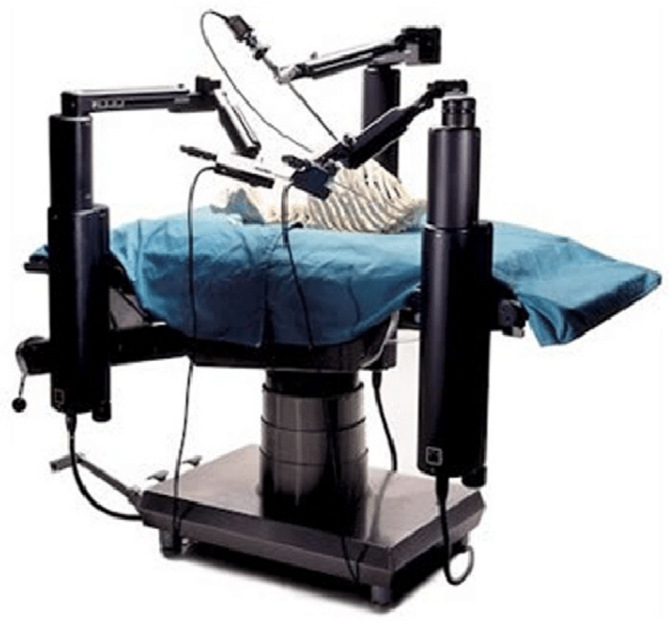


Figure 6. ZEUS robotic surgical system by Computer Motion. Extracted from: https://www.researchgate.net/figure/ZEUS-robotic-system-first-robotic-system-to-combine-instrument-and-camera-control_fig3_51437277

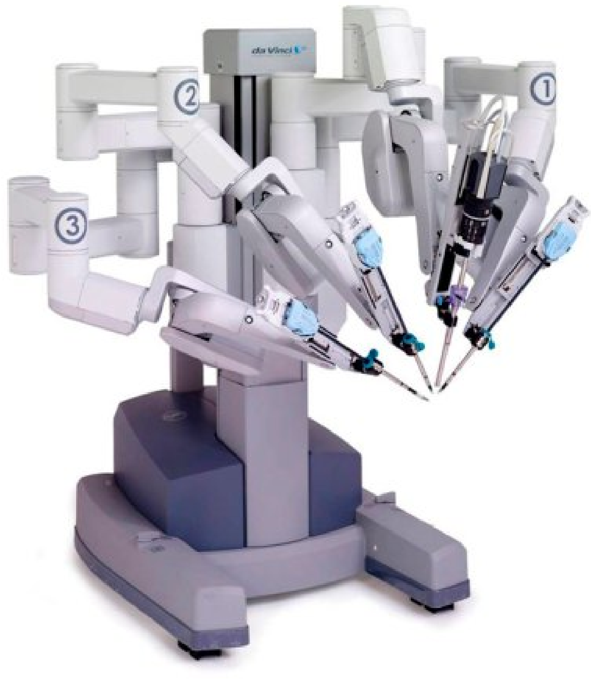


Figure 7. Da Vinci surgical robotic system. Extracted from: https://blogthinkbig.com/robot-quirurgico-da-vinci

4. Engineering of conception

The DaVinci surgical system has some limitations as previously mentioned. Nonetheless, the most critical one is the lack of tactile sensitivity. Therefore, the main reason why this project started was to solve this above-mentioned limitation.

Here, there will be exposed the necessary tools that will be needed for this project as well as a study between several options for determining which fits best our necessities. The study will be divided in two main sections: one for the hardware, which includes the one needed to control the robotic arm, the motors, the haptic user interface, among others; and the software.

An overview of the project main goal is necessary to understand the proposed solutions and the ones chosen. This work consists of implementing haptic feedback in a robotic arm with a DaVinci EndoWrist tool. Therefore, a robotic arm with 6 degrees of freedom is required. Moreover, to simulate the movement of the suture performed by the surgeon, a system to control the position and orientation is also essential. Eventually, a system to translate this movement into the end-effector of the robotic arm, which would be the needle, is required as well. Besides this hardware setup, a stable software needs to be used for the overall control of the system.

4.1. Hardware solutions

In order to carry out this project, several hardware devices are going to be needed. In this section robotic arms and haptic feedback devices will be exposed and therefore analyzed.

To choose the best option, several factors must be taken into account such as cost and their technical pros and cons.

4.1.1. Robotic arm

The human arm has 7 degrees of freedom (DoF) which includes pitch, yaw and roll between the shoulder, the elbow and the wrist. Nonetheless, the da Vinci surgical system has 6 DoF [16]. Since the EndoWrist instruments will be appended to the new robotic arm through a designed device and they already have some degrees of freedom, the robotic arm won't need as many DoF.

There are different options when it comes to acquire a new robotic arm. It can be built up from zero or it can be purchased. While producing it from zero would seem cheaper and better when it comes to design it the way we want it, it is not as feasible since there isn't enough time nor technical knowledge. On the contrary, if an already commercialized robot is used the difficulty and time decreases and also, we can find more available programming languages.

Here, some robotic arms will be exposed and compared regarding their degrees of freedom and price (see Table 1):

Model	Company	Degrees of freedom	Cost (€)
UR5e [17]	Universal Robots	6	30.890 [18]
Dorna [19]	Dorna	5	1.324
Robolink [19]	Igus	4-5	5.560
Magician [19]	DOBOT	4	1.060

Table 1. Robotic arms comparison.

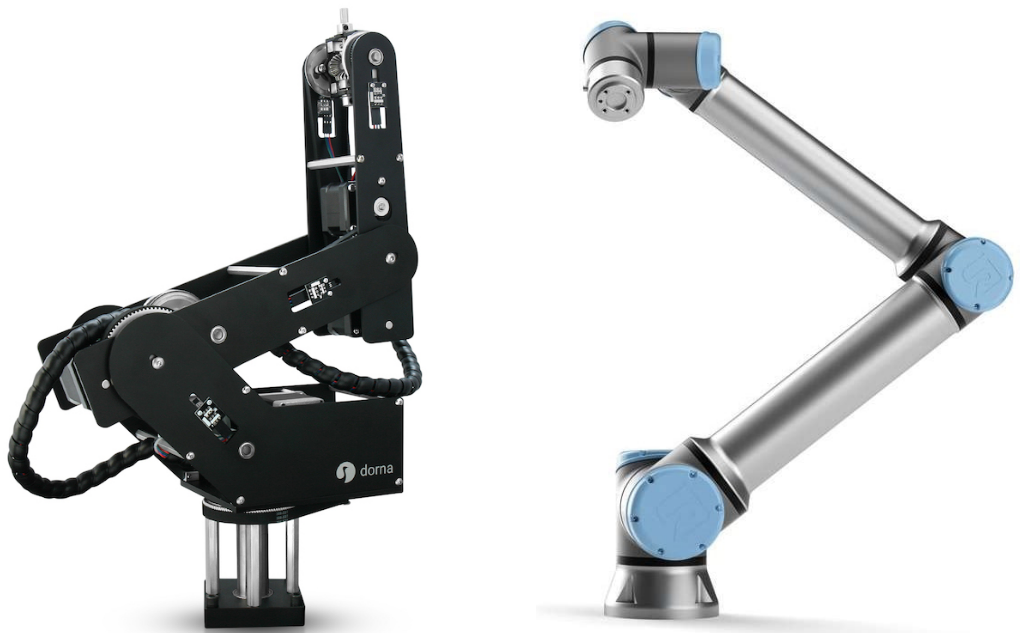


Figure 8. (Left) Dorna model. (Right) UR5e robotic arm. Extracted from: https://dorna.ai/wp-content/uploads/2019/02/Image7-1.jpg & https://wiredworkers.io/product/ur10/

4.1.2. Haptic feedback user interface

Since the main aim of this project is to integrate haptic feedback into a surgical robotic arm, a haptic feedback system that provides this tactile sensitivity must be selected. Therefore, in this section, several devices will be studied.
In order to choose one, there are some factors that need to be evaluated such as cost, degrees of freedom, its nominal position and its stiffness.
Here, we can see some available devices in the market and some of their specifications (see Table 2):

Model	Position resolution (mm)	Mean stiffness (N/mm) (x, y, z)
Touch [20]	0'055	1'26 – 2'31 – 1'02
Touch X [20]	0'023	1'86 – 2'35 – 1'48
Sigma.7 [21]	0'0015	-
Omega.7 [21]	0.006	14'5

Table 2. Haptic controllers comparison

With these features we would choose the ones with the lowest resolution and the highest stiffness. Nonetheless, they are much more expensive.
Besides these options, another possibility would be to develop a de novo haptic system by using different sensorial stimulus such as vibration or sound. If this option was selected, an inertial mass unit sensor would be needed as well.

4.1.3. Inertial Mass Unit sensor

In the case a new system is developed and none of the previous haptic feedbacks are used, a sensor to measure the orientation and angles of the hand movement is required.
For this purpose, several sensors are being proposed in the following table (see Table 3).

Model	Axis	Power	Cost (€)
Sensor 2019 Adafruit Industries	3	3V, plus a 3.3V LDO regulator and level shifting circuity	8'41
MPU-9250/6500	9	3 – 5 V (with internal regulator)	12'99

Table 3. Inertial Mass Unit sensors comparison

These Inertial Mass Unit sensor mentioned in the table above may be found in the following websites:

- **Sensor 2019 Adafruit Industries**: https://es.rs-online.com/web/p/circuitos-integrados-de-sensores-de-movimiento/9054665/
- **MPU – 9250/6500:** https://www.amazon.com/-/es/MPU-9250-6500-aceleraci%C3%B3n-giroscopio-magnet%C3%B3metro/dp/B07QN6V56Z

4.1.4. Haptic stimulus

Regarding the haptic feedback, it's the stimulus the surgeon will receive as soon as he or she surpasses a determined force.
For this, several stimulus have been studied. On the one hand, a resounding signal could be used, whose sounding frequency varied depending on how much the force surpassed the threshold. For this implementation, a buzzer would be a suitable option. The following table shows and compares different models (see Table 4):

Model	**Frequency**	**Intensity**	**Voltage**	**Price**	**Buzzer**
AI-2604-TF-LW115-12V-R	400 ± 100 Hz	89 dB at 12 V	8 V ~ 16 V	2,92 €	
Piezoelectric buzzer RS PRO	2,9 – 3,9 kHz	95 dB	3 V ~ 20 V	2,53 €	
MCKPT-G1720-3922	Up to 4 kHz	85 dB	Up to 30 V	1,07 €	

VMA319	1,5 – 2,5 kHz	-	5 V	4,95 €	

Table 4. Buzzer comparison for haptic stimulus

The buzzers shown in the previous table may be bought in the following websites:

- **AI-2604-TF-LW115-12V-R:** https://www.digikey.es/product-detail/es/pui-audio-inc/AI-2604-TF-LW115-12V-R/668-1347-ND/1745449
- **Piezoelectric buzzer RS PRO**: https://es.rs-online.com/web/p/componentes-de-piezo-buzzer/7541999/
- **MCKPT-G1720-3922:** https://es.farnell.com/multicomp/mckpt-g1720-3922/piezo-buzzer/dp/1756525
- **VMA319:** https://www.planetaelectronico.com/vma319-modulo-arduino-zumbador-activo-p-16146.html

On the other hand, vibration may be a valid haptic stimulus as well. As in the previous option, vibration intensity could vary depending on the torque surpassed. In Table 5, a study of different vibration motors may be seen:

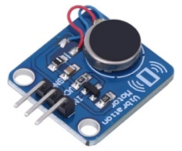
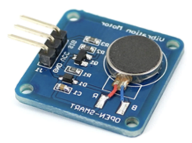
Model	Voltage	Price	
Mini vibration motor Seeed Studio	2,5 – 3,5 V	1,23 €	
DC5V 9000RPM	3 – 5,3 V	7,59 €	
Mini vibration motor Open – Smart	3,0 – 5,3 V	1,06 €	

Table 5. Comparison of vibration motor for tactile haptic feedback

These vibration motors may be bought in the following websites:

- **Mini vibration motor Seeed Studio:** https://www.robotshop.com/es/es/motor-disco-vibrante-mini-2mm.html?gclid=Cj0KCQjw8IaGBhCHARIsAGIRRYrPI7nEAQMHtsEAE61xUgJHjsL0-XdQtva_4cGvHj43e75KH4z91pQaAmvfEALw_wcB
- **DC5V 9000RPM:** https://www.amazon.es/vibraci%C3%B3n-9000RPM-Sensores-proyectos-bricolaje/dp/B088FV1R8V/ref=asc_df_B088FV1R8V/?tag=googshopes-21&linkCode=df0&hvadid=495635487753&hvpos=&hvnetw=g&hvrand=13196185031692

311762&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=1005424&hvtargid=pla-1071368078620&psc=1
- **Mini vibration motor Open – Smart**: https://es.aliexpress.com/item/32677263318.html

Another option could be implementing both devices to get a more intense feedback. In this case, it would be highly interesting they both are able to work at the same voltage.

4.1.5. Servo motors

Servo motors are crucial for the translation of the force exerted by the surgeon into the torque needed to rotate the engines of the EndoWrist. Moreover, they allow a position feedback since most of them contain a sensor [22]. Four servo motors will be needed: one for the roll, another one for the pitch and two for the yaw to perform grasping.
It's of high importance the resolution, the maximum torque the servo motor allows and the speed of rotation. Given that the final implementation is intended to be in a surgical environment, the accuracy of rotation, orientation and position plus the total control of the force exerted are key therefore, the parameters evaluated for the decision will include angle resolution and torque.
As previously mentioned, the servo motors will transmit the surgeon's force into the EndoWrist engines. Since this device will require from an input signal, it may be assumed that the force exerted by the surgeon will be proportional to the servo motor's torque. It therefore, would be of great importance if the servo allowed to extract the current supplied or the torque implemented.
Several options are proposed in this section (see Table 4). The Dynamixel AX-12 servo motor proposed by Arturo [23] in the previous version of this project is considered as well.

Servo Motor Model	Company	Angle Resolution (degrees, º)	Stall torque (N·m)	Load feedback
Dynamixel AX-12	Robotics	0.29	1.5	Yes
SG-90 Micro Servo	Longruner	0.3	0.098	Yes
HS Ultra Torque	Hitech	0.23	0.36	No
Servomotor R/C	Parallax Inc	-	0.27	Yes
FT1117M-FB	FEETECH	-	0.34	Yes

Table 4. Servo motor models comparison

For the purpose of this project it's of high importance the load feedback to be able of reading the torque exerted by the motor when rotating.

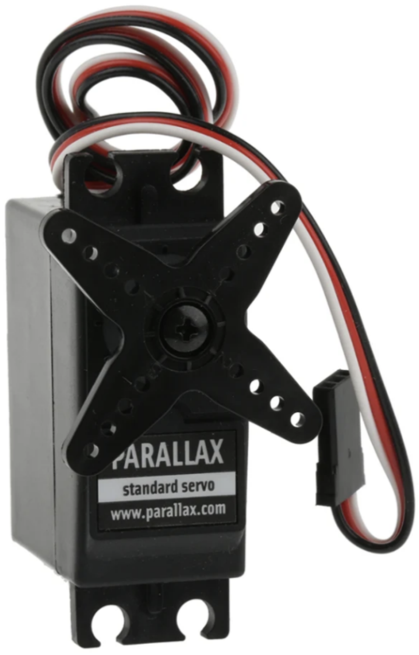


Figure 9. Servomotor R/C by Parallax Inc.

For further information and to buy any of the previous servo motors, one may check the following websites:

- **Dynamixel AX-12**: https://ro-botica.com/Producto/Actuador-Dynamixel-AX-12A/
- **SG-90 Micro Servo**: https://www.amazon.es/Servo-Motor-Control-Helic%C3%B3ptero-LKY66-UK-10/dp/B07236KYVC?th=1
- **HS Ultra Torque**: https://hitecrcd.com/products/servos/micro-and-mini-servos/digital-micro-and-mini-servos/hs-5070mh-ultra-torque-metal-gear-feather-servo/product
- **Servomotor R/C:** https://es.rs-online.com/web/p/motores-dc/7813058/?cm_mmc=ES-PLA-DS3A-_-google-_-PLA_ES_ES_Automatizaci%C3%B3n_y_Control_de_Procesos_Whoop-_-(ES:Whoop!)+Motores+DC-_-7813058&matchtype=&aud-821594433763:pla-478037629623&gclid=Cj0KCQjwwLKFBhDPARIsAPzPi-lTZhh7hG1QtWSarfMdeDhaYxLWrzxqBCRgLQLjTXqe1Lnpht3Vd6waAvU-EALw_wcB&gclsrc=aw.ds
- **FT1117M-FB**: https://tienda.bricogeek.com/servomotores/1320-mini-servo-feetech-3-5kg-ft1117m-fb-con-feedback.html?gclid=Cj0KCQjwwLKFBhDPARIsAPzPi-KKJwkKntEVWGmXcvU3QfcTQ0d0M_qe2Czx_kqC_lmHlE6hy3x1j2kaAph0EALw_wcB

4.1.6. Control hardware

In order to control the servo motors as well as the Inertial Mass Unit sensor if it is eventually necessary, it's going to be needed a hardware board (see Table 5).

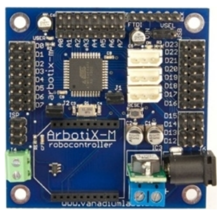
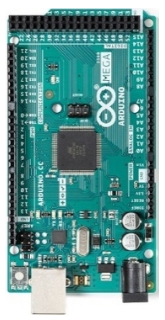
Model	Supported Software	Cost (€)	Board
Open CM 9.04 C	Arduino IDE and OpenCM IDE [24]	20'95	
Arbotix – M Robocontroller	Arduino IDE [25]	35'30	
Servocontrolador Raspberry Pi Dynamixel	Arduino IDE and ROS	79'95	
Dynamixel Shield	Arduino IDE	28	
NUCLEO – F401RE	STM32 and Arduino IDE	16'39	
Arduino Mega	Arduino	41'26	

Table 5. Hardware boards comparison

The aforementioned hardware boards may be bought in the following websites:

- **Open CM 9.04 C**: https://www.mybotshop.de/OpenCM-904-C-ROBOTIS
- **Arbotix – M Robocontroller** : https://www.trossenrobotics.com/p/arbotix-robot-controller.aspx
- **Servocontrolador Raspberry Pi Dynamixel**: https://www.mybotshop.de/Servocontrolador-Raspberry-Pi-Dynamixel
- **Dynamixel Shield**: https://www.ro-botica.com/Producto/DYNAMIXEL-Shield/
- **NUCELO – F401RE**: https://es.farnell.com/stmicroelectronics/nucleo-f401re/placa-nucleo-mcu/dp/2394223?gross_price=true&CMP=AFC-CJ-ES2394223
- **Arduino Mega**: https://es.rs-online.com/web/p/arduino/7154084/?cm_mmc=ES-PLA-DS3A-_-google-_-CSS_ES_ES_Raspberry_Pi_%26_Arduino_y_M%C3%B3dulos_de_Desarrollo_Whoop-_-(ES:Whoop!)+Arduino-_-7154084&matchtype=&aud-821594433763:pla-369227342804&gclid=Cj0KCQjwwLKFBhDPARIsAPzPi-I7yjczLwIcfjAcP85dWwD8YHd8sc7NAsY-Wif1TpGoakaju8d3_IlaAm_-EALw_wcB&gclsrc=aw.ds

4.2. Software solutions

For this project not only the hardware is important but also the software since it's necessary to control the robot. In this section there will be exposed and discussed several software (all compatible with Arduino IDE) that may be used. Besides the software to control the whole system's performance, different 3D and PCB designing software are discussed as well since many pieces have been designed and printed by Meiling Chen, who has also participated in the development of this project both in the mechanical and electronic part.

4.2.1. Programming software

In order to control the robot a software is needed. As it can be seen in the following table (see Table 6), we can consider these software:

Software	Compatibility with Arduino	Cost (€)
ROS	Yes	Open source (Free)
Python	Yes	Open source (Free)
LabView	Yes	406'00/year [26]
MATLAB & Simulink	Yes	800/year [27]

Table 6. *Software comparison*

They are all high-level programming languages which are programming languages with strong abstraction from the details of the computer and allow the programmer to be detached and separated from the machine. Moreover, they can execute functions which are already programmed nonetheless it may have a constraint: we cannot fully control the device. For this, we should use low-level programming languages, which require higher programming skills [28].
While ROS and Python are open sources which means they are free, LabView and MATLAB & Simulink have a high cost. Nonetheless, since we are students from Universitat de Barcelona, we may get a student license.

4.2.2. 3D designing software

This project is being carried out in collaboration with the EUSS school. Meiling Chen is the student who has been helping out both in the mechanical and electronic part.
For the mechanics of the system, several components are needed. Given the requirements needed, the best option is to design and print them ourselves since finding the specific pieces with the characteristics desired on the market may be extremely difficult.
For the 3D design of the pieces there are loads of programs available nonetheless, some of them require from a license which may be costly. To decide which program will be used, a brief comparison will be exposed.

Software	Context	Knowledge and skills	Price
SolidWorks	Mechanical, industrial and medical	Normal	$1,295/year
AutoCAD	Gold standard in industrial sectors	It requires from an extensive training	$210/month
Inventor	Many	Highly advanced	$340/month
FreeCAD	Many	High level not required	Free

***Table 7**. Comparative table of 3D designing software.*

While SolidWorks is a solid modeling computer-aided design (CAD) and computer-aided engineering (CAE) 3D modeling software, AutoCAD may be used for both 2D and 3D computer-aided design (CAD) and the fact that has loads of features makes this software very versatile and useful for many industrial sectors. Furthermore, it has loads of features that make this software very versatile and useful for many industrial sectors. Moreover, some tools may help automatize the designs. A little drawback is that it requires from an extensive training in order to have a good command of the software.
Inventor, besides requiring from highly advance skills, it's highly recommended for users that require accurate 3D designs, visualizing and simulation capabilities.

On the other hand, FreeCAD doesn't require from a high level of knowledge but it isn't as intuitive as the others. It's an open-source, highly customizable, scriptable and extensible. Moreover, since it's built on Python, new features may be implemented by programming them [29].

When considering which software to use, price is an important factor. We therefore need to see if there are available student licenses or free trials. In case it isn't possible, we could always choose a free software. Nonetheless, it is important to keep in mind that it will have less features and tools and it won't be as easy to use nor intuitive as the pay-to-use ones.

4.2.3. PCB designing software

For the assembly of the final prototype, a printed circuit board (PCB) may be the most suitable option since it would optimize not only space but also the connections will be much better than if they were welded or connected through wires. PCBs are the basis of any hardware-based product and, before the existence of software, engineers designed these electronics circuitry and integrated circuits manually. As in any other type of software, there are both free and premium versions. While the most popular ones contain more tools and are not free, there are still some strong free programs which are the ones that will be exposed and compared in this section.
This part is done by Meiling Chen as well, who is majoring in mechanical and electronics engineering.

Software [30]	Advantages	Operating system
KiCAD	Designs with up to 32 copper layers.	Windows, Mac and Linux
Fritzing	Includes a breadboard layout and a PCB view.	Windows, Mac and Linux
DesignSpark PCB	Includes a schematic, a PCB editor and allows unlimited number of layers.	Windows

Table 8. *Comparative table of PCB designing software.*

KiCAD is entirely free and no paywall is required for extra features, as well as Fritzing and DesignSpark PCB. Nonetheless, the fact that this last one is only available for Windows, makes it a less appealing solution.

4.3. Proposed solution

This section will expose the options that have been chosen after the study of the previous section. Regarding the hardware of our prototype, the robotic arm is not going to be built up from zero since it would take too long and we don't have enough technical skills. Therefore, after analyzing the options to purchase, we have eventually chosen the UR5e robotic arm from Universal Robots since it is already in Universitat de Barcelona and it is available at Biomedical Engineering and Electronics Department of the Physics Faculty.

Regarding the haptic controller user interface, we have decided to build it from zero. Thanks to the participation of Meiling Chen in this project, a 3D pencil will be printed. This pencil will include an Inertial Mass Unit sensor, which will control the servomotors that will rotate according to the orientation of the IMU.

For the development of this controller, the sensor chosen is the IMU model MPU-9250/6500 since it moves along 9 axis, which allows more degrees of freedom, it may be supplied between 3 and 5 V, it's already available at Universitat de Barcelona and we already know how it works and how to acquire its orientation and send it to the servo motors with Arduino IDE.

Regarding the servo motors, SG-90 Micro Servo is the model eventually chosen due to its small size, the low stall torque and because it allows reading the load feedback, which is essential for the main goal of this project. Moreover, as it happens with the IMU model, these servos have already been used in previous projects therefore, we already know how they work, how to connect them and their specifications.

Concerning the hardware to control both the sensor and the servo motors we will opt for two models depending on the stage. For the trials, NUCLEO – F401RE will be used due to it has already been used in previous projects and it's cheaper than Arduino Mega, which will be the one used for the final prototype. This final change is given by the fact that Arduino Mega may be supplied at both 5V and 12V and allows an external supply, which isn't possible with NUCLEO – F401RE. Since we need to supply four servomotors, a sensor, a buzzer and a vibration motor, an external supply is the best option to play it safe. Moreover, to connect the four servo motors a shield compatible with Arduino Mega will be used.

For the haptic feedback, we will opt for using two types of stimulus: auditive and tactile. For the auditive one, the buzzer VMA319 will be used and the frequency will change depending on how much the torque surpasses the threshold. Regarding the tactile stimulus, a vibration motor used on mobile phones will be used. As in the buzzer, the delay between vibrations will decrease as the torque increases above the threshold.

Regarding the software, since we are using the Arduino Mega board and it's compatible with Arduino IDE we need a software capable of integrating it. Although all the proposed options allow this, we will finally choose ROS because besides being free, it's open source, which means that there are loads of libraries and functions available provided by other users. Moreover, it is a software specific for robotics, it may be programmed in various languages and allows peer-to-peer communication, which helps avoiding communication problems in complex robotic systems that have multiple links [31].

Not only the haptic feedback user interface will require 3D printing but also other mechanical parts of the system such as a piece that will include the four servo motors. All these components, which will be designed and printed by Meiling Chen, need 3D designing software. SolidWorks is the option

chosen since it's the one Meiling knows best and it provides loads of libraries and tools that are extremely useful and helpful when designing industrial and mechanical components.
For the PCB design, KiCAD is the software eventually chosen since its performance is good, it provides loads of tools and also because it's the software Meiling Chen has more knowledge about.
Therefore, this is the final proposed solution:

- Hardware:
  - UR5e by Universal Robots
  - MPU – 9250/6500 Inertial Mass Unit sensor
  - SG-90 Micro Servo
  - NUCLEO F401RE
  - Arduino Mega
  - Shield board
  - Buzzer VMA319
  - Vibration motor
  - Haptic user interface with custom made *pen – haptic.*

- Software:
  - Arduino IDE
  - ROS
  - SolidWorks
  - KiCAD

4.3.1. Overall prototype conception

For a better understanding of the previous section, a schematic of the overall prototype conception is exposed in the following figure (see Figure 10).

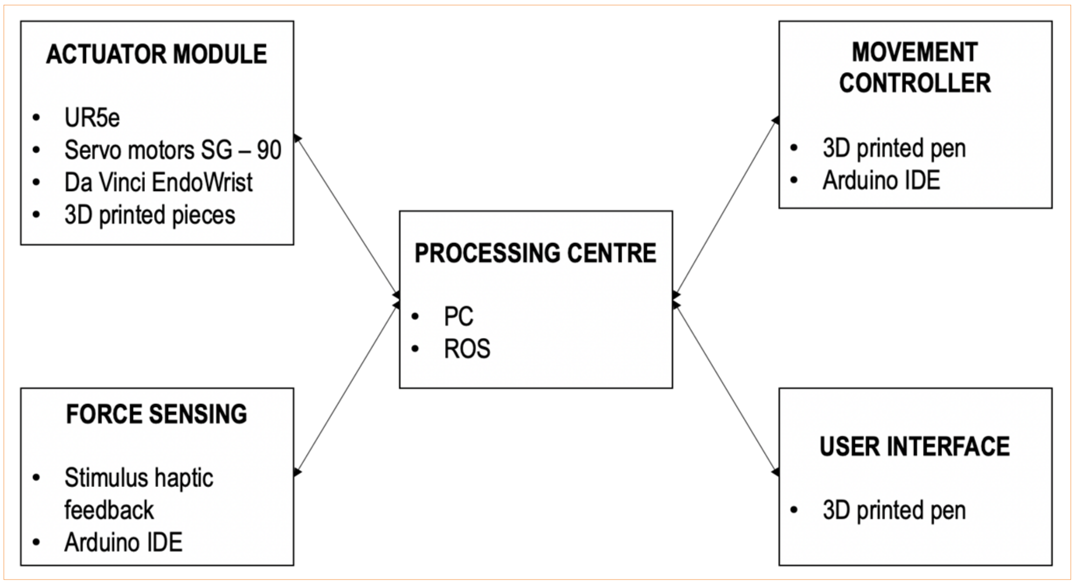


Figure 10. Schematic of the overall prototype conception made by the author of this document

When talking about the *user interface*, one is referring to the part the user would interact with and would manipulate. In this case, the surgeon would only use the 3D printed pen to simulate the suture process.

For the *actuator module* not only the UR5e and the Da Vinci tool are needed, but also the servo motors, since are the ones that will translate the movement into the end – effector. Moreover, the final assembly of all the components will be encompassed by different 3D printed pieces.

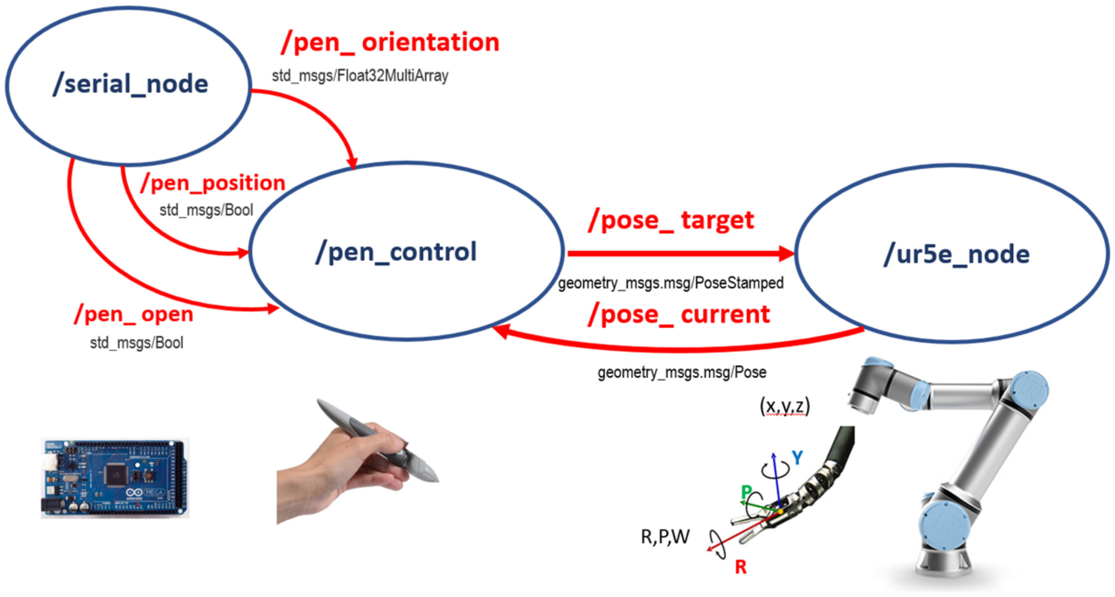


Figure 11. Schematic of the communications between hardware. Made by Dr. Manel Puig Vidal

For the final prototype, the idea is to use two hardware boards: one for the robotic arm UR5e and the other one for the haptic pen.

The haptic pen will include the IMU and will capture the orientation and position of it. Moreover, it will include the buzzer, vibration motor and push button as well. The IMU orientation will be sent via WIFI to the other hardware board, the one connected to the robotic arm and will move the corresponding servos. Nonetheless, the communication is bidirectional since the servos will read the torque, which will be sent to the haptic pen and activate the buzzer and vibration motor in case it surpasses the threshold. This may be clearly seen in Figure 11.

5. Detailed engineering

In this section, it will be explained the whole process of design and implementation of this project. Moreover, all the considerations that have been taken into consideration for the correct assembly and the tests that have been performed will be exposed as well.

5.1. Hardware implementation

One thing to be considered is that this project is constituted by two different blocks (see Figure 12) and thus, two different hardware and software programs.

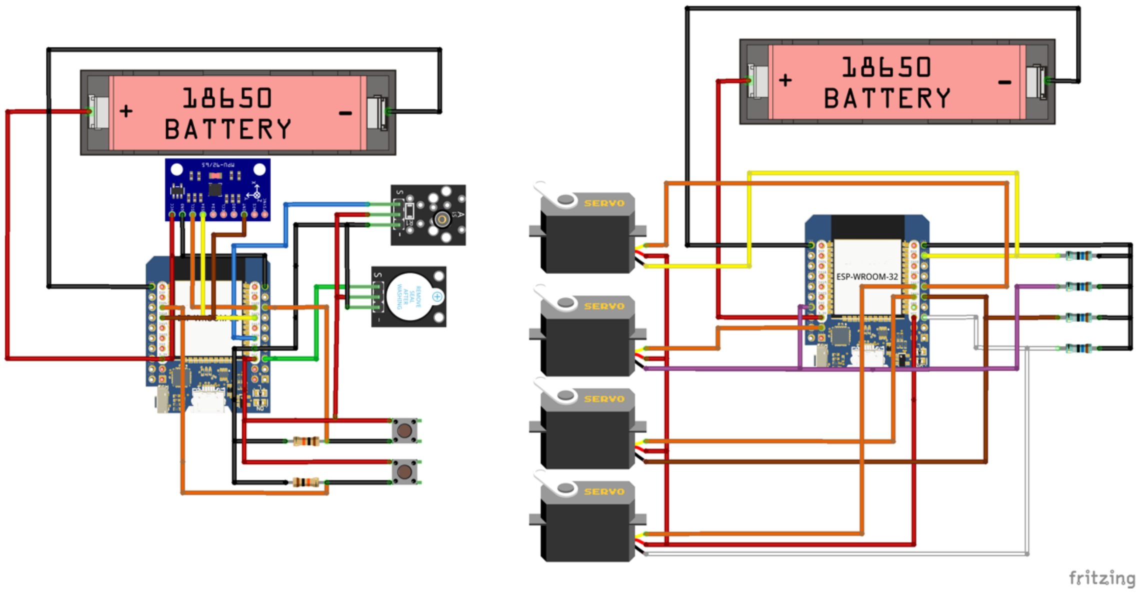


Figure 12. Schematic of the two hardware blocks done by Meiling Chen.

The first block is the haptic pen, which is manipulated by the surgeon and contains the IMU and the corresponding actuators. It captures the hand movement and sends the angles to the second block, which is the UR5e arm with the servo motors. These servos will rotate the indicated angles and will measure the torque exerted. This torque will be sent again to the aforementioned hardware, that will activate the actuators in case the exerted force surpasses a threshold.

5.1.1. Haptic pen hardware

As it has been mentioned in the previous section "Engineering of Conception" for this project it has been chosen to build a haptic pen device instead of using an already available haptic system. The 3D design has been done by Meiling Chen. The following image shows the final design, with the assembling of the sensors and actuators.

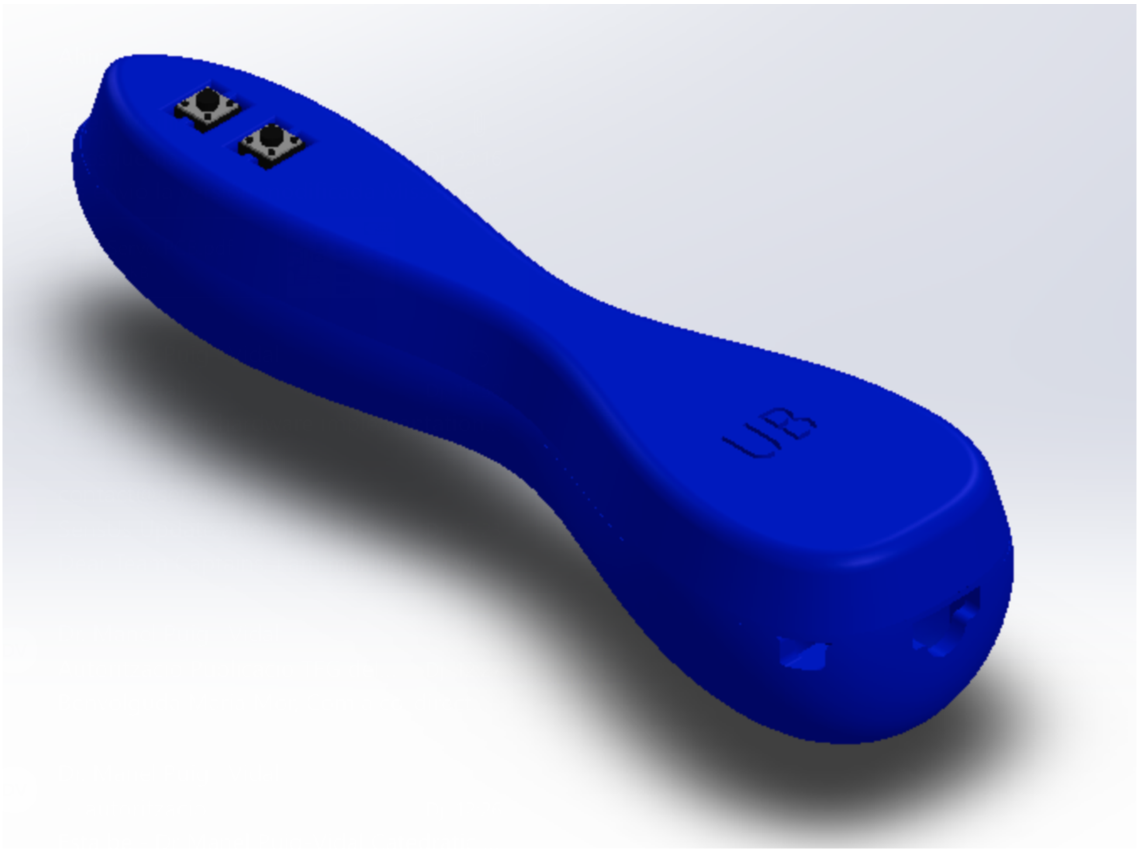


Figure 13. Haptic pen designed by Meiling Chen

For the correct functioning of the pen several tasks have been performed.

5.1.1.1. Inertial Mass Unit sensor implementation

In the first place, the assembling and implementation of the Inertial Mass Unit sensor is key since it is responsible of simulating the hand movement.
Therefore, the first step of this project has been to assemble the IMU MPU – 9250/6500 to the NUCLEO F401RE.

As mentioned in "Engineering of Conception", the software used is Arduino. Thus, the orientation and position of the IMU sensor is read by an Arduino code available at the GitHub repository of Albert Álvarez Carulla.

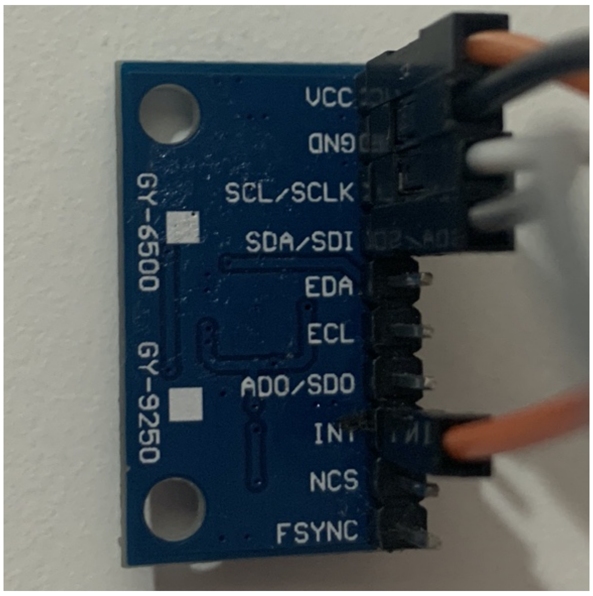


Figure 14. Pin connections of the IMU sensor. Made by the author of this document

The pins are connected to NUCLEO F401RE the following way:

- VCC to 3V3
- GND to GND
- SCL/SCLK to SCL/D15
- SDA/SDI to SDA/D14
- INT to D8

More information about the Arduino code may be found in the Annexes of this document.

For the final prototype, the IMU sensor will be located inside the haptic pen together with the buzzer and vibration motor.

5.1.1.2. Reading the torques of the servos with an external resistance with the IMU

After ensuring the IMU read correctly the angles, the accurate movement of the servos as well as the reading of the torque was key to achieve the main goal of this project.
By knowing the torque of the servos at a constant speed without any force exerted on them and comparing it to the torque when an opposed force is applied, is vital to determine the threshold.

Figure X shows the assembly of a single servo, which shows an external resistance as well. This resistance has a value of 1'6 Ω and it's key to measure the torque, which would be expressed in millivolts.

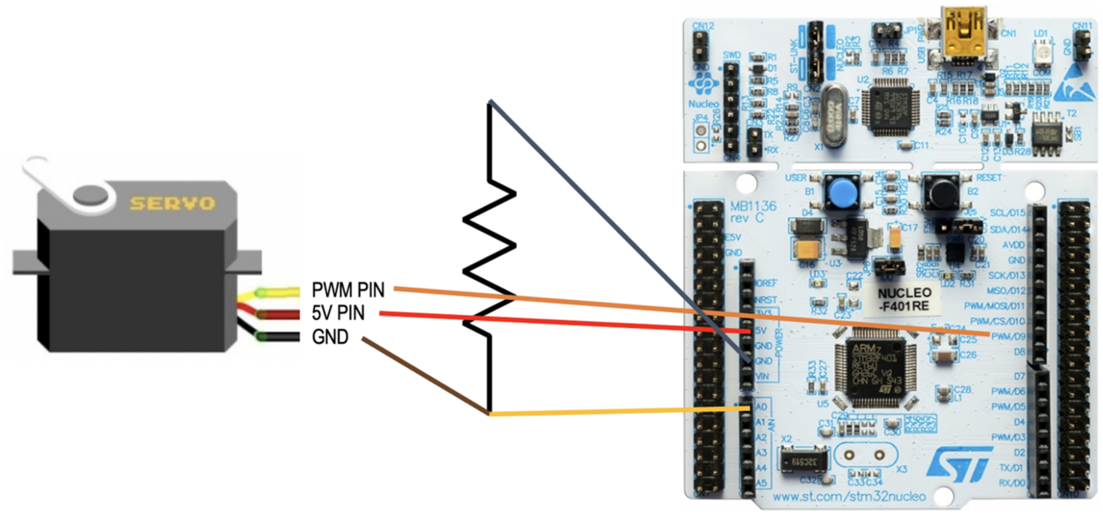


Figure 15. Assembly of a single servo motor to the hardware board. Made by the author of this document

Nonetheless, when testing the results it was seen that the torque was not constant when no force was applied and therefore, no reliable threshold could be stablished. To solve this, a digital filter was applied to smooth the signal via integration. The time constant of this filter, which is eventually 200 ms, was determined by trial and error.

After checking the correct movement of a single servo with the IMU and its correct torque measurement, we proceeded to assemble the four servos (see Figure 16).
Each IMU angle had to correspond to one servo, except the yaw movement which had to control two servo motors. Therefore, the Arduino code had to be readapted to move each servo a determined angle. For a deeper insight of the code, you may check the Annexes of this document.

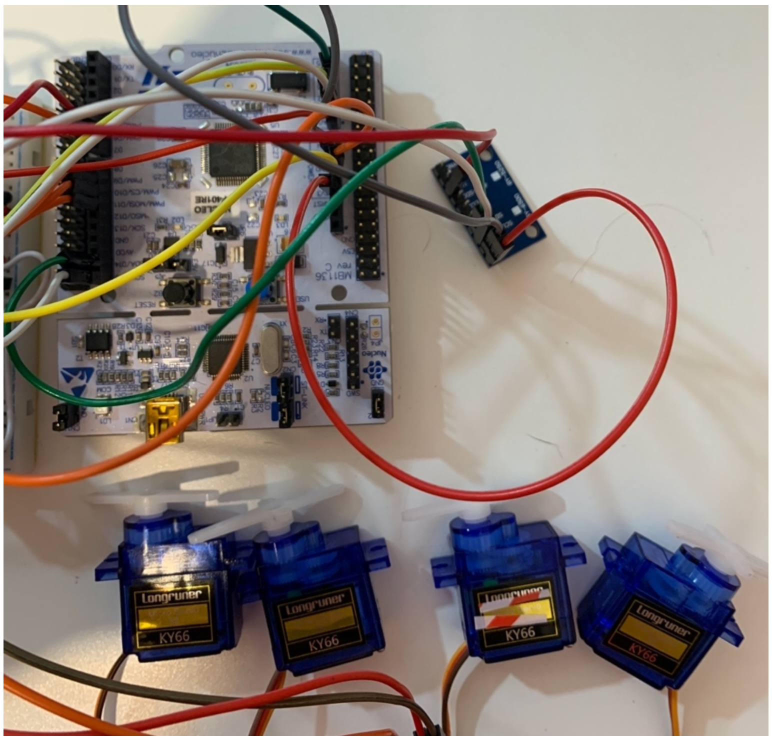


Figure 16. Assembly of the four servo motors which move according to the IMU sensor

5.1.1.3. Implementation of the buzzer and vibration motor

Both the buzzer and vibration motor are the actuators that will provide haptic feedback once the torque surpasses a determined threshold.
They will be inside the haptic pen as seen in Figure 17.

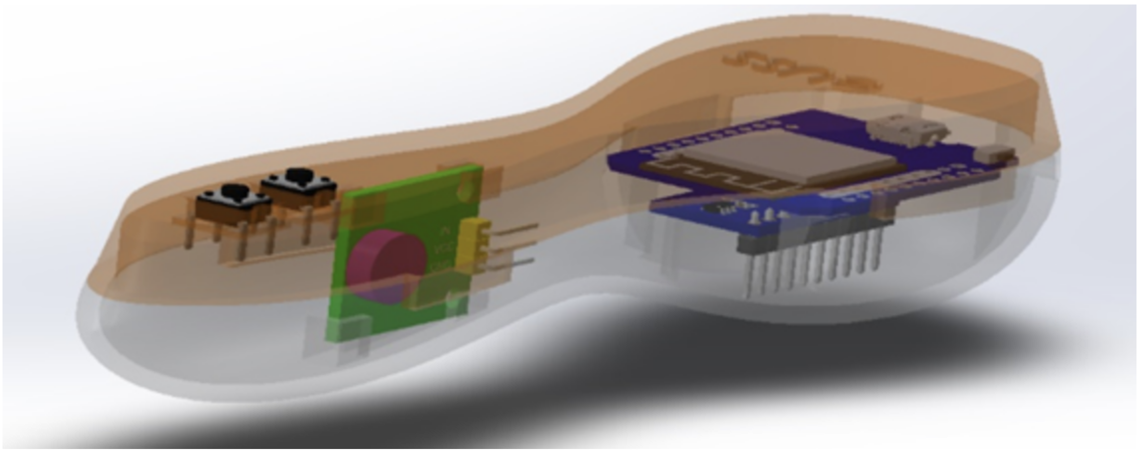


Figure 17. Location of buzzer and vibration motor in the haptic pen. Made by Meiling Chen

For a more accurate perception of the force exerted by the surgeon, we proposed that both the frequency and intensity of the buzzer and the vibration motor increased as the force raised too. This has been done by applying a rule of three in the parameters of the Arduino code, which may be seen in the Annexes as well.

5.1.2. UR5e arm hardware

As previously mentioned, the final prototype is composed by two hardware blocks. The second one encompasses the UR5e arm, which includes the servo motors and the EndoWrist.

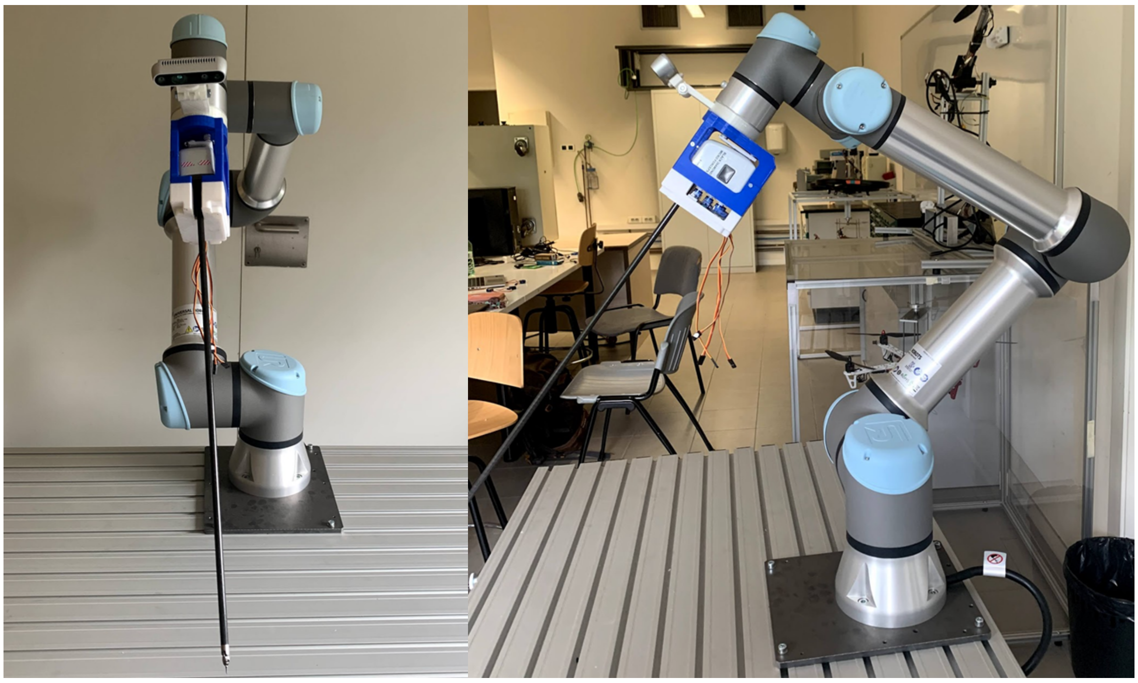


Figure 18. UR5e arm assembly with the EndoWrist frontal (left) and lateral (right). Made by the author of this document.

This module is the one that will be in touch with the patient. The EndoWrist will reproduce the surgeon's hand movement according to the rotation of the servo motors that will move given the orientation of the IMU sensor located in the haptic pen. This communication between the two hardware is done by Meiling Chen using ESP–32.
Considering that the robotic arm will be subjected to movements and forces, a strong and consistent union between the robotic frame, the servo motors and the Da Vinci EndoWrist end-effector has been done using 3D printed pieces designed by Meiling Chen as well.
The white part seen in Figure X will contain the top of the Da Vinci EndoWrist tool.

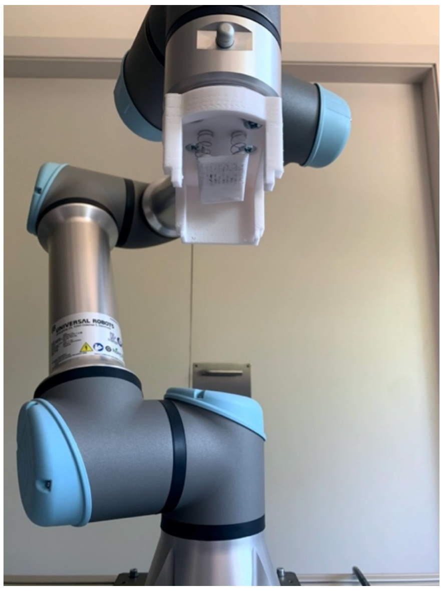


Figure 19. Component for the assembly of the UR5e to the EndoWrist tool. Made by the author of this document

On the other hand, the servo motors will be connected to the tool by using a gear assembly system.

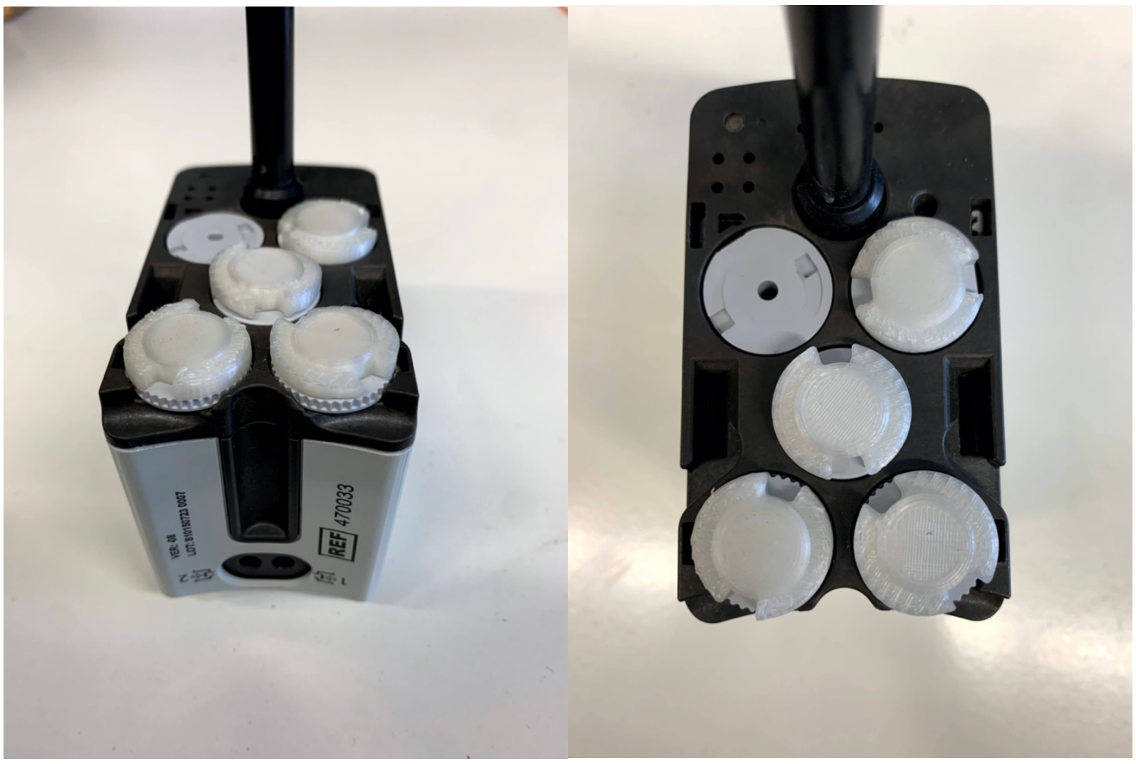


Figure 20. Pieces for the gear assembly with the servo motors. Made by the author of this document.

5.1.2.1. Reading the torques of the four servos with an external resistance without the IMU

Given that the IMU and the servo motors will be connected to different hardware, it has been necessary to connect them independently.
Although the connections are the same as in the previous section, it is worth mentioning that the Arduino code varies a little given that this time the servo motors will rotate a determined angles which is indicated in the code as a variable.
For further detail about the code implemented, one may check on the annexes of the document.

5.2. Experimental validation

To properly work and develop the final prototype, all the previously exposed assemblies have to be tested.
The main programs used have been Arduino to program and control the hardware devices and visualize and print data such as torque values; and RoboDK for simulation purposes.

5.2.1. Validation of the IMU sensor in RoboDK

To check if the IMU sensor faithfully captured its orientation a simulation with RoboDK was performed. To do this, we had to establish serial communication between Arduino and RoboDK. The code may be seen in the annexes of this document.
It was verified the correct functioning of the IMU sensor through the movement of the UR5e simulation in RoboDK and the RPY angle values printed in the Tkinter.

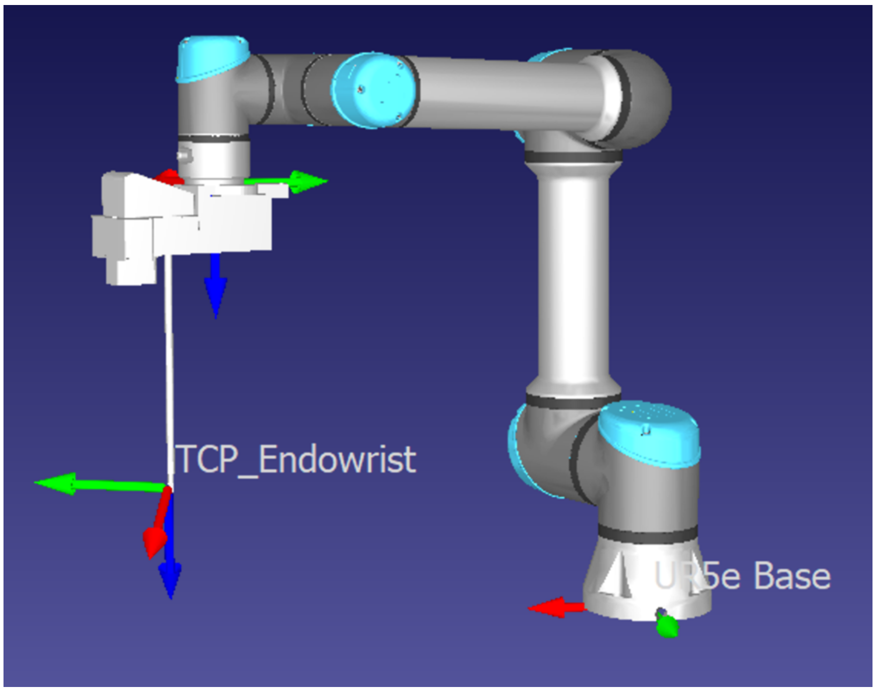


Figure 21. UR5e simulation in RoboDK. Made by the author of this document.

The *TCP_Endowrist* shown in the image moved according to the IMU. We could therefore verify and accept the results.

5.2.2. Validation of the torque

For the validation of the torque as well as to stablish a threshold for the vibration motor and buzzer, several tests have been performed regarding the torque of the servo motors.
To test our components, both the serial plotter as well as the monitor series tools of Arduino have been used to visualize the outcome.
The following figures show the difference in torque between the first case, where no contrary force was exerted to the servo; and the second case, in which an obstacle was situated in front of the propeller and therefore, complicated the spin and increased the torque.

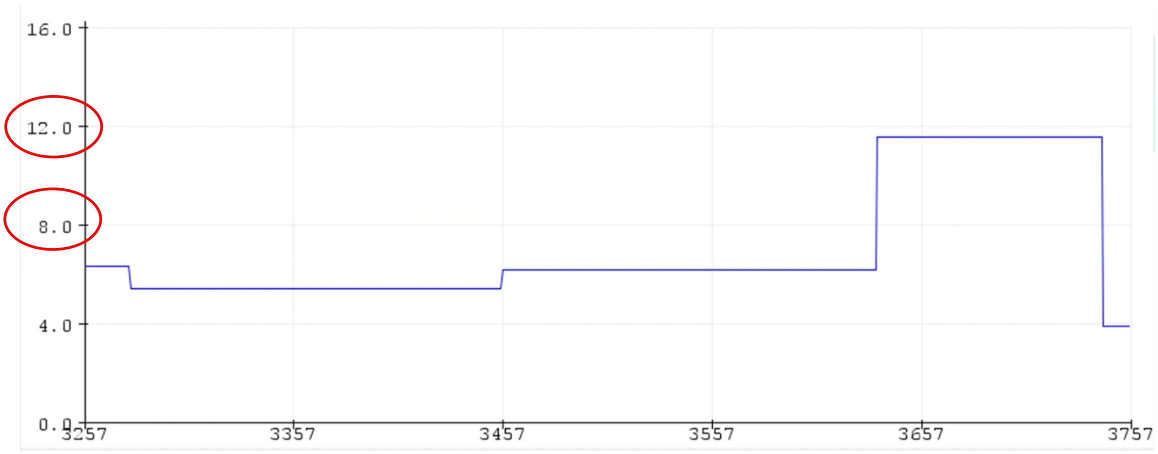


Figure 22. Torque of one servo motor without resistance

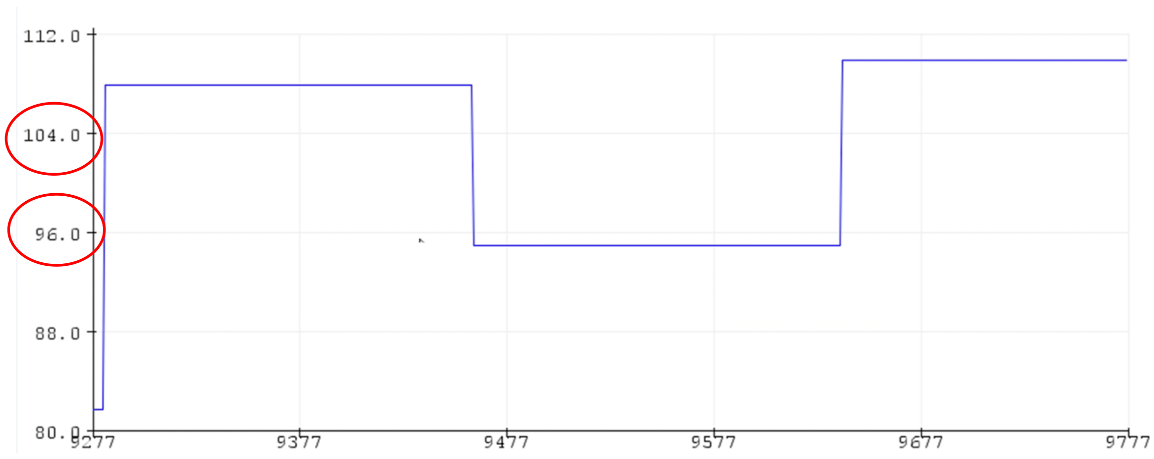


Figure 23. Torque of one servo motor exerting resistance

Comparing both figures and their Y axis, it’s clearly seen how the torque between the two cases differ. Given this outcome, we could stablish an approximate value of torque as a threshold: 100.

Although the difference between torques regarding the force applied is significant, there are still visible steps that should ideally be much smaller. To solve this we tried changing the time constant of the digital filter up to 500 ms and 1000 ms. The results may be seen in the following figures.

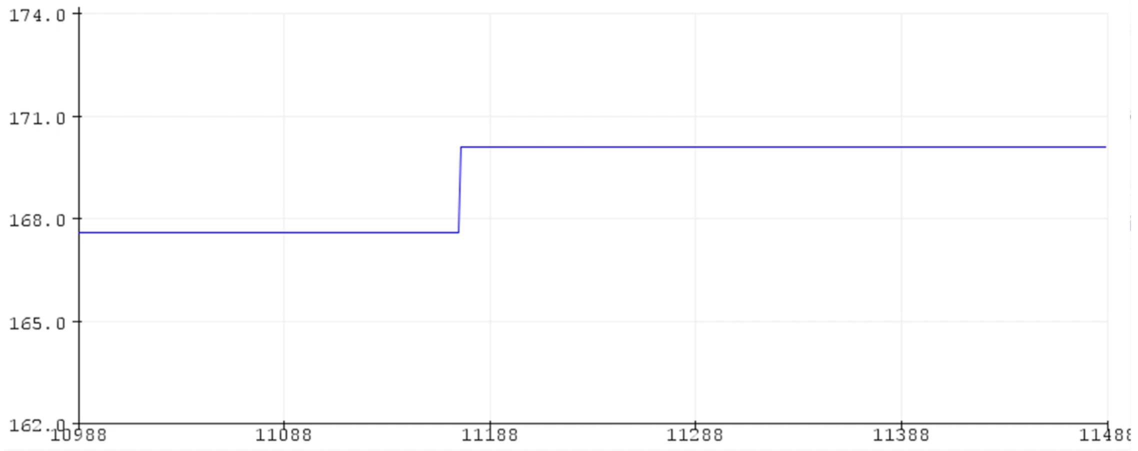


Figure 24. Torque of one servomotor exerting external force with constant time equal to 500 ms

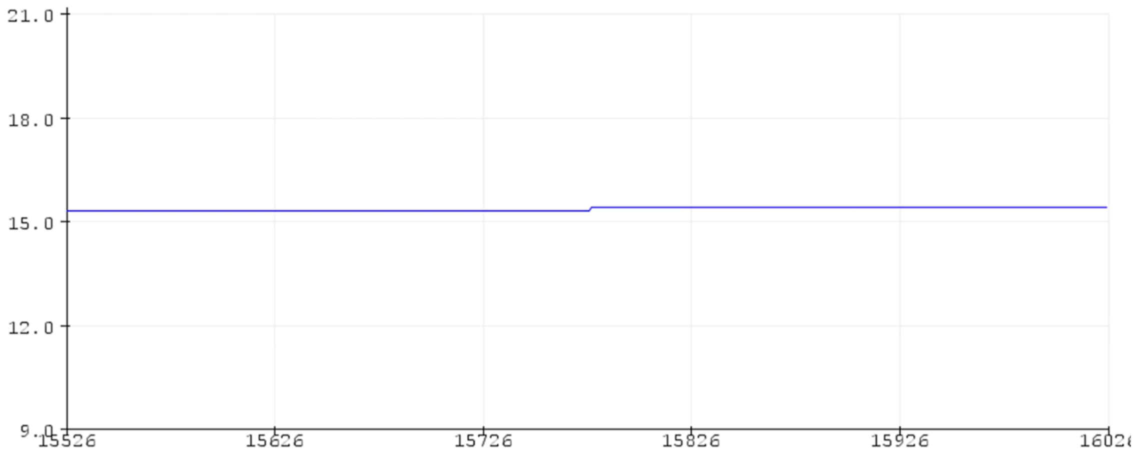


Figure 25. Torque of one servomotor without external force with constant time equal to 500 ms

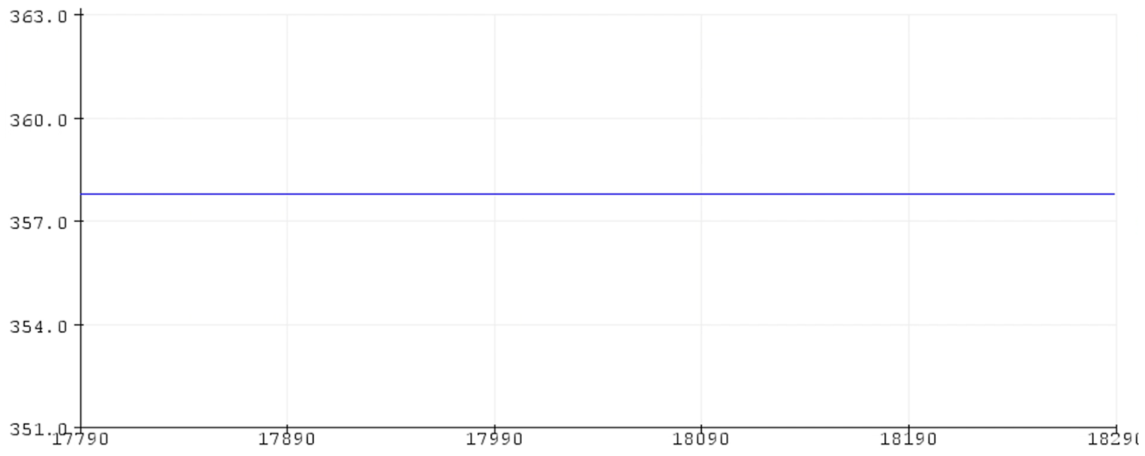


Figure 26. Torque of one servomotor exerting external force with constant time equal to 1000 ms

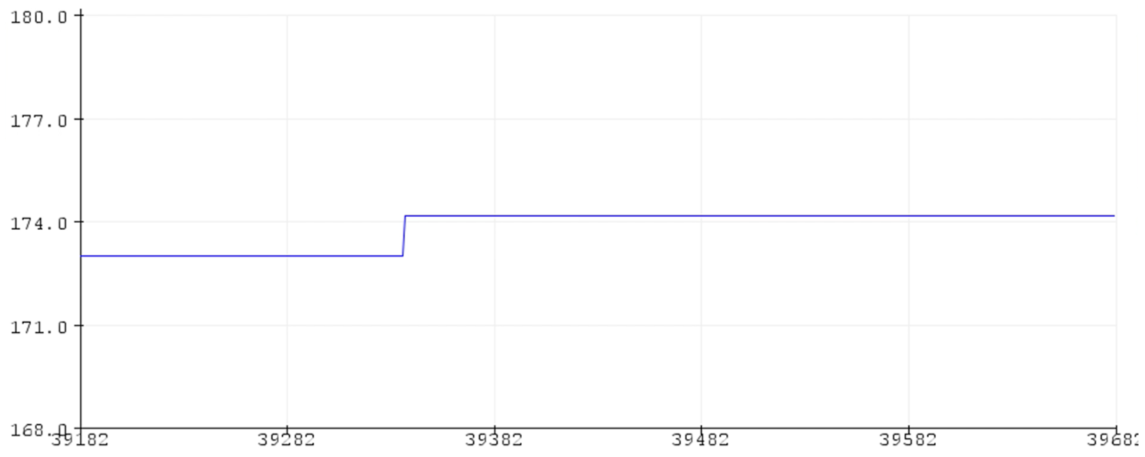


Figure 27. Torque of one servomotor without external force with constant time equal to 1000 ms

Although the graph may look smoother, the time delay was too considerable and could affect negatively the surgeon's performance since the actuators would activate later. Moreover, it may be seen how the higher the time constant is, the higher the torque is. This may be due to the sensitivity of the torque.

Another solution we proposed was implementing a capacitor as a RC filter. Given the following expression (see Equation 1):

$$\tau = R \cdot C$$

Eq 1. Time constant

Nonetheless, the value of the capacitor was too high given the small value of the resistance. Since changing the resistance was not an option because the servos wouldn't work, this option was eventually declined.

Because of all the previously mentioned reasons plus the fact that the torque variance wasn't as considerable as it seemed, we decided to accept the results given by a time constant of 200 ms.

5.2.3. Buzzer and vibration motor validation

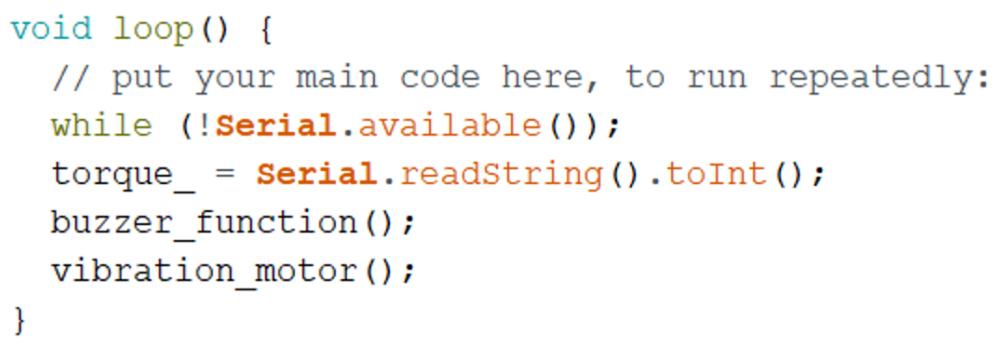
To validate the buzzer and the vibration motor, several tests have been performed.

In the first place, a simple condition was established by using an “If” function in Arduino in both cases separately. A torque threshold was established and in case the torque variable was higher than the threshold, the vibration motor and the buzzer should activate.

After this simple test, we thought it could be interesting to increase the frequency and intensity of noise and vibration as the torque increased.

To check this a “for” loop was designed in both cases so that different torque values were given increasingly. This way we verified the correct functioning of both actuators.

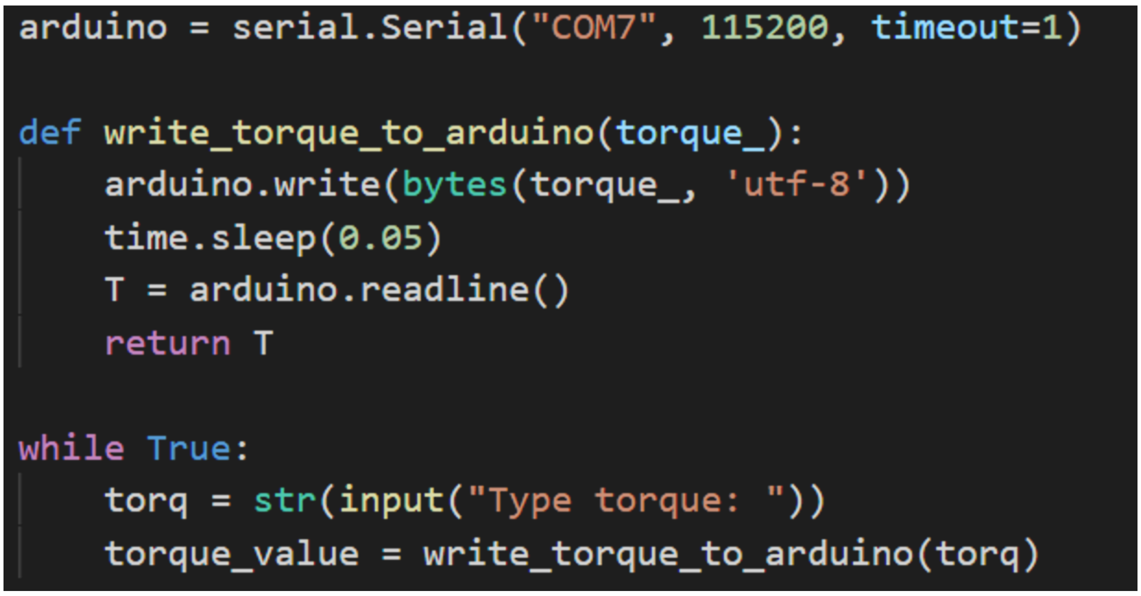
Eventually, we decided to put both actuators together so that they would activate at the same time.



```
void loop() {
  // put your main code here, to run repeatedly:
  while (!Serial.available());
  torque_ = Serial.readString().toInt();
  buzzer_function();
  vibration_motor();
}
```

Figure 28. Integration of buzzer and vibration motor with torque value from RoboDK.

Given that the torque value will be sent from another hardware block, we decided it would be more accurate if the torque was sent from another program to Arduino to validate their functioning. Therefore, a serial communication from RoboDK to Arduino was established and the variable torque was sent from the RoboDK software.



```
arduino = serial.Serial("COM7", 115200, timeout=1)

def write_torque_to_arduino(torque_):
    arduino.write(bytes(torque_, 'utf-8'))
    time.sleep(0.05)
    T = arduino.readline()
    return T

while True:
    torq = str(input("Type torque: "))
    torque_value = write_torque_to_arduino(torq)
```

Figure 29. Serial communication from Python in RoboDK to Arduino.

It was eventually verified the correct performance of both actuators, which activated as soon as a torque higher than one hundred was sent from RoboDK and accentuated their behaviour as the torque increased and stopped as soon as the torque became lower than one hundred again.

For further information about the programming codes, one may check the annexes of this document.

6. Regulations and legal aspects

Since this
project focuses on a medical device which will be directly in contact with patients it must follow strict regulations and technical aspects to be approved by the FDA or obtain the CE stamp here in the European Union.
Due to this project is about the same medical device as previous research projects [23] the legislation will be extremely similar. However, it has been updated to the newest legislation [32].

6.1. Hardware of medical devices

The main regulations this type of medical devices must follow in order to guarantee their safety, quality and efficiency are:

- ISO 13485:2016. It specifies the requirements for a quality management system.
- ISO 10993-1:2018. Biological evaluation of medical devices – Part 1: Evaluation and testing within a risk management process.
- ISO 14971:2019. Medical devices – Application of risk management to medical devices.
- ISO 15223. Medical devices – Symbols to be used with medical device labels, labelling and information to be supplied.

6.2. Software of medical devices

When it comes to the software, we find other regulations:

- IEC 62304. Medical device software – Software life cycle processes

6.3. Sterilization and disinfection

Since we are talking about a RMIS device it means that it has to get into the surgery room and has to be in contact with the patient. Therefore, it must follow strict processes of sterilization and disinfection whose protocols and requirements are described in the following regulations: ISO 14937, ISO 17664:2017 , ISO 11135, ISO 7153-1:2017, UNE-EN ISO 11137-3:2018 and ISO 11138.

6.4. Safety requirements and risk management

Risk management is key in the development of medical devices due to the fact that patients are already very susceptible. It's important to ensure their health and safety during the whole process of diagnosis and treatment and protect them from risks that could affect them.
ISO 14971 [33] is the standard that specifies the process so the manufacturer can easily identify the hazards, estimate, evaluate and control the risks associated and monitor the quality and effectiveness of the tests.
UNE – EN IEC 60601 [34] is for medical electrical equipment. This standard establishes the requirements needed for ensuring the safety and good performance of devices with power sources. Since the Da Vinci system has some electrical currents for cutting and coagulating, this regulation needs to be considered.
Besides ISO 14971, UNE – EN 62366 [35] is necessary as well, since it specifies the application of usability engineering to medical devices. An important part of risk prevention is the proper utilization of the instrument.

6.5. Protection of the data

Like other device or service that works with patients, data privacy is fundamental. ISO 27701 [36] and ISO 27001 are the international standards that cover data and information privacy and security concerns.

6.6. Symbols and visual indications

It’s of high importance both the correct labelling of the medical devices with symbols and the accurate information provided by the manufacturer, which are considered in ISO 15223 and ISO 20417 [37], respectively.

7. Technical viability

In this section, the strengths, weaknesses, opportunities and threats of this project are being presented. They are also known as internal and external factors. Knowing them is important in order to be conscious of what our project offers in front of the rival companies and also to improve our performance by boosting the positive factors and fixing the negative ones.

7.1. Intern analysis

7.1.1. Strengths

As it has been already seen in the previous sections, the lack of haptic feedback in surgical robotic systems is a huge constraint whom none of the companies available nowadays in the market has solved yet. According to several studies, it has been demonstrated that tactile perception may improve the surgeon's intervention.

Another strength this project presents in front of other companies is its low cost. While the da Vinci system costs around $2 million our prototype is way cheaper.

7.1.2. Weaknesses

This prototype is not allowed to get inside a surgery room at the moment. It must pass loads of healthcare regulations and fulfil several legal aspects such as CE and FDA approval. This processes may take a long time and some money, which can make the prototype to become more expensive.

Another weakness that must be considered is the lack of time, some resources and technical knowledge about the performance of the da Vinci. Although it has been analyzed and studied through bibliographic research, it's highly protected by loads of patents and confidential clauses.

7.2. Extern analysis

7.2.1. Threats

As it has been seen in "Analysis of the market", there are loads of companies in the RMIS market and although there isn't any which has implemented haptic feedback yet, there're already some which are already making certifications and getting ready to launch their project to the market.

Moreover, the da Vinci System will turn 20 years this year which means that some of their documents won't be confidential anymore. This will lead to an increase in competitors in the RMIS field.

7.2.2. Opportunities

In "Engineering of conception" we've seen that there is a wide of options to choose between when it comes to both hardware and software. This allows us to choose according to our interests and the performance they may provide us.

Although this project has focused on implementing haptic feedback, other factors could be improved and considered for future lines. Reducing the time delay between the hand movement and the end effector and improving the vision and perception of the surgeon could be considered as future projects. Therefore, this study offers a wide range of new opportunities since it’s scalable to other improvements.

7.3. SWOT analysis

After exposing the strengths, weaknesses, opportunities and threats this project has, future lines can be considered and also helps to be aware of the limitations and problems one will need to deal

with when carrying out this project. Although there are several weaknesses and threats, they can be considered as challenges for the future.
For a better understanding, the following table shows what has been previously mentioned (see Table 9):

	Favorable	Unfavorable
Interna	Strengths ○ Implementation of haptic feedback in RMIS ○ Low – cost	Weaknesses ○ Too many certifications and approvals ○ Difficult access to the da Vinci System's information
Externa	Opportunities ○ Technology advance and availability ○ Scalable to other challenges	Threats ○ New prototypes of different companies ○ New competitors

Table 9. *SWOT analysis table*

8. Execution chronogram

In this section, the necessary activities to achieve the project goals are defined.

Defining the tasks and milestones is crucial for the project's planning and therefore knowing the resources and time needed (some extra time has been considered in some tasks in case a problem came up). Thus, key activities must be defined by elaborating an activities' matrix. This way, it can be graphically represented with PERT and GANNT diagrams.

8.1. Definition of tasks and timing

In the following figure it can be seen the work – breakdown structure (WBS) of this project (see Figure 30):

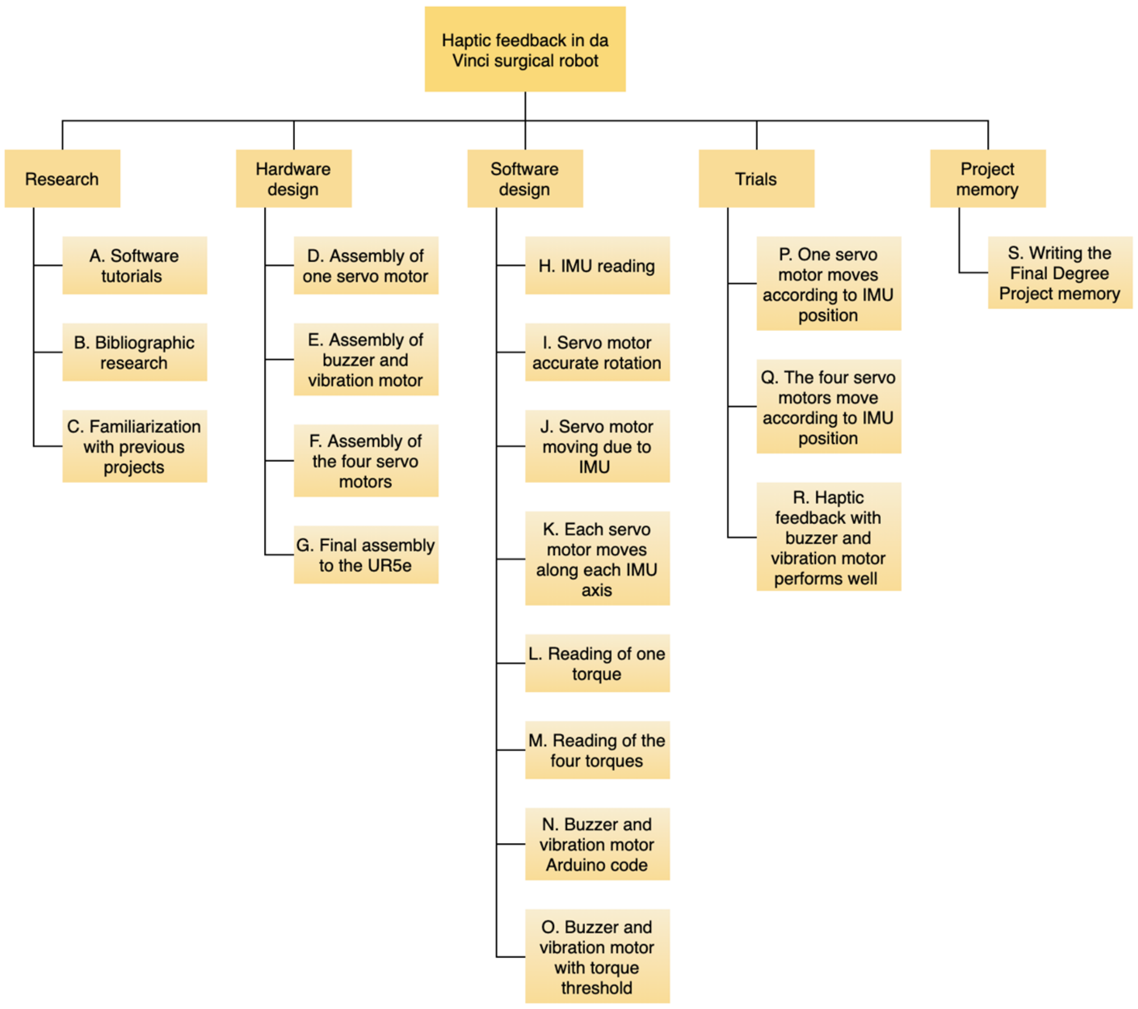

Haptic feedback in da Vinci surgical robot
Research
Hardware design
Software design
Trials
Project memory
A. Software tutorials
B. Bibliographic research
C. Familiarization with previous projects
D. Assembly of one servo motor
E. Assembly of buzzer and vibration motor
F. Assembly of the four servo motors
G. Final assembly to the UR5e
H. IMU reading
I. Servo motor accurate rotation
J. Servo motor moving due to IMU
K. Each servo motor moves along each IMU axis
L. Reading of one torque
M. Reading of the four torques
N. Buzzer and vibration motor Arduino code
O. Buzzer and vibration motor with torque threshold
P. One servo motor moves according to IMU position
Q. The four servo motors move according to IMU position
R. Haptic feedback with buzzer and vibration motor performs well
S. Writing the Final Degree Project memory

Figure 30. *Work – breakdown structure of the project*

With this we can estimate how long it will take to complete each task (see Table 10):

ACTIVITY	PRECURSOR	TIME (in weeks)
A	-	18
B	-	18
C	-	18
D	B, C	4
E	B,C	4
F	D	6
G	F, O	2
H	C	4
I	C	4
J	H, I	3
K	J	4
L	C	4
M	L	5
N	C	2
O	M, N	7
P	J	4
Q	K	4
R	O	1
S	B, C, G, R	20

***Table 10**. Project tasks with their durations and predecessors*

8.2. GANTT chart

In this subsection the GANTT chart is shown. This type of diagram is a bar chart which illustrates the schedule of a project and shows the dependency between the activities (see Figure 31).

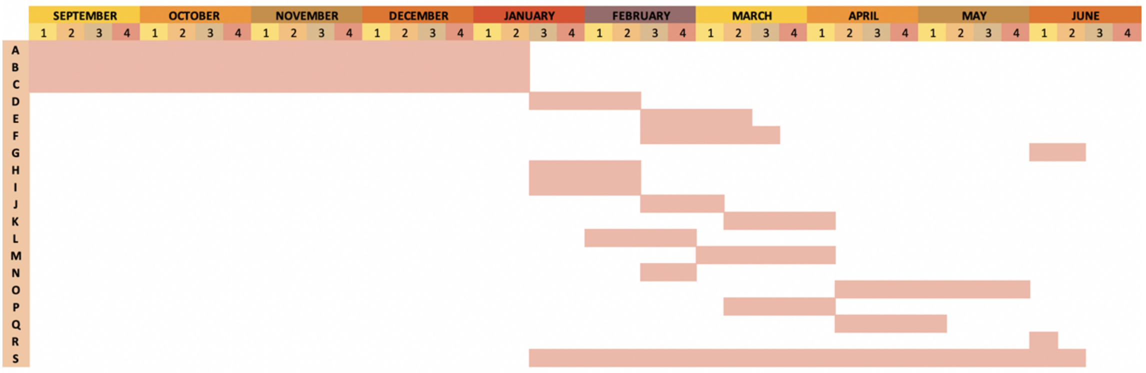


Figure 31. GANTT chart of the Project

According to this Gantt diagram, this project will take up to 38 weeks.

9. Economic viability

This section will encompass all those factors that will suppose an economic cost during the project. Since this is a draft we cannot totally ensure the final budget. Nonetheless, it will be the most accurate possible.

In here, not only the tangible material will be considered but also the intangible one such as, software licenses and the professionals' salary (see Table 11).

Material	Units	Cost per unit	Total
Biomedical engineer	1	15€/h · 300h	4.500€
ROS	1	Free	-
Arduino IDE	1	Free	-
KiCAD	1	Free	-
SolidWorks	1	Student license 119'79€/year	Financed by EUSS
Computer	1	1.500€	1.500€
UR5e	1	30.890€	30.890€
Inertial Mass Unit sensor	1	12'99€	12'99€
Arduino Mega	1	41'26€	41'26€
NUCLEO F401RE	1	16'39€	16'39€
Shield board	1	7'28€	7'28€
Servo motor	4	13'99€ (pack of 5 servo motors)	13'99€
Buzzer	1	4'95€ (pack of 2 buzzers)	4'95€
Vibration motor	1	1'06€	1'06 €
TOTAL			36.987'92€

Table 11. *Total budget of the project*

Taking into account the previous table, the total cost of this project is about 36.987'92€.

10. Discussion and conclusions

In this section it will be discussed both the fulfillment of the initially planned objectives and the future lines and improvements this project may encompass in further studies.

10.1. Objectives fulfillment

In this subsection, the objectives defined at the beginning of this project and considered in the execution chronogram will be analyzed and in case of not being fulfilled, it will be explained in detail the reasons behind as well as some weaknesses and flaws in order to keep in mind for further studies.

- Regarding the study of the da Vinci's surgical performance it has been successfully achieved by bibliographic research, attending a surgery at Hospital Clinic while studying *Aplicacions Mèdiques de l'Enginyeria 3*, and by testing and analyzing the EndoWrist available at Universitat de Barcelona laboratory. Thanks to this tool, it has been possible to study the in detail the maneuverability of the end – effector and how the servos should rotate in order to accurately simulate the surgeon's hand movements.

- Evaluating the software eventually programmed, it has been achieved a partially suitable integration and communication between the haptic pen and RoboDK. The haptic actuators, which include the buzzer and vibration motor activate properly as soon as the torque indicated from RoboDK surpasses the established threshold. Nonetheless, it has not been possible to simulate at the same time the movement of the UR5e arm at RoboDK while using the IMU sensor. Despite this, we have been able to simulate it separately, which was one of the main objectives as well. Although the final integration has not been total, the software program of the haptic feedback has been successful.
  On the other hand, the evaluation of the torque has been exhaustive and validated through different methods and tests. First with the IMU sensor and one servo motor, then with four servo motors and finally without the IMU. Moreover, different time constants have been considered for the digital filter and also the use of a RC filter was studied but, although the graphs of the torque may seem smoother the time delay is too considerable and may suppose a risk for the patient.

- Regarding the hardware stage we have been able to assembly the four servo motors into the da Vinci EndoWrist which is connected to the UR5e arm as well. The servo motors where tested with the IMU but the tests with ROS and the communication between the haptic pen and the robotic arm have not been eventually performed due to lack of time.
  On the other hand, although the idea was to assembly the final prototype using an Arduino Mega and a Shield board for the servo motors, eventually only the NUCLEO R401RE has been used due to lack of materials. For the assembly of the servos into the Shield board, a specific type of connector, which wasn't available at the laboratory, was needed and there wasn't enough time to purchase it.

10.2. Future lines

In this section, some improvements that could be implemented in future projects have been described.

Nonetheless, it has to be considered that the final prototype should be able to be used in a surgery room, which means that loads of improvements regarding legislation, sterilization, accuracy, non-toxicity among other factors, should be done.

Given the unfulfilled goals of this project, future studies could try to solve them. Regarding the software, a final program able to read the IMU, reproduce the movements in the RoboDK and, at the same time, send torque values to activate the actuators should be done.

On the other hand, an enhancement of the hardware used is also recommended. For future projects, the prototype could be assembled in an Arduino Mega and the servo motors in a Shield board.

Regarding the final validations, verifications and tests, the programs with ROS should be implemented, as well as the communication by Wi-Fi between both hardware blocks.

Besides the implementation of haptic feedback, surgical robots present other limitations that could be studied in future projects. One of them could be the enhancement of vision since the surgeon may have a distorted perspective and may not perceive accurately the distances between tissues and the EndoWrist. A system that included a sensor to indicate the proximity could be an interesting future line.

10.3. Personal conclusions

As has been already mentioned several times during this project, the main objective was to design, develop and integrate a haptic feedback prototype to implement into a robotic arm for surgery purposes.

Although the idea was for medical applications, it was never an objective nor was the intention to eventually implement this prototype into a currently available robotically assisted medical system given that medical devices are highly regulated and need to pass loads of tests, certifications and criteria.

It’s worth mentioning that the fact that this project was inherited from previous projects, allowed a deeper investigation and knowledge about the topic. Moreover, studying what previous people have done may provide us with new perspectives and ideas that may result in better outcomes. This is what Isaac Newton meant when saying “If I’ve seen further, it is by standing on the shoulders of giants”.

This project has been a great opportunity to learn not only technical skills such as programming with Arduino or Python but also with loads of soft skills like time and task organization as well as teamwork. Working together with Meiling Chen and Julia Meca has been a pleasure and has enriched my mind. Moreover, it has been a relief to have such great teammates whenever an issue could appear.

Last but not least, I find it necessary to mention the importance of the advances in both the technological and medical fields. This has been seen during this Covid-19 pandemic, where thanks to medical devices and vaccines we have been able to face it.
It’s of high importance to keep investigating and improving in these fields, this is why we decided to start and continue this project that focuses on enhancing surgical procedures by trying to solve the lack of tactile perception by implementing haptic feedback in robotically assisted surgical systems.

11. References

[1] St. Peter, S. and W. Holcomb, G., 2009. History Of Minimally Invasive Surgery.
[2] En.wikipedia.org. 2020. Da Vinci Surgical System. [online] Available at: <https://en.wikipedia.org/wiki/Da_Vinci_Surgical_System> [Accessed 30 May 2020].
[3] Es.wikipedia.org. 2020. Sistema Quirúrgico Da Vinci. [online] Available at: <https://es.wikipedia.org/wiki/Sistema_quir%C3%BArgico_Da_Vinci> [Accessed 30 May 2020].
[4] Meddeviceonline.com. Da Vinci S Surgical System Enables Complex Surgery Using Minimally Invasive Approach. [online] Available at: <https://www.meddeviceonline.com/doc/da-vinci-s-surgical-system-enables-complex-su-0001> [Accessed 30 May 2020].
[5] Davincisurgerycommunity.com. Da Vinci Si/Si-E. [online] Available at: <https://www.davincisurgerycommunity.com/Systems_I_A/da_Vinci_Si_Si_e> [Accessed 30 May 2020].
[6] Intuitive.com. 2020. [online] Available at: <https://www.intuitive.com/en-us/products-and-services/da-vinci/systems##> [Accessed 30 May 2020].
[7] Abexsl.es. Abex. [online] Available at: <http://www.abexsl.es/en/robot-da-vinci/da-vinci-xi> [Accessed 30 May 2020].
[8] Cirugía Robótica San Rafael. 2017. Robot Da Vinci X: Primera Prostatectomía Radical En España | Cirugía Robótica San Rafael. [online] Available at: <https://cirugiaroboticasanrafael.com/archivos/1025> [Accessed 30 May 2020].
[9] N. Chavan, E. and V. Rojarkar, D., 2017. State Of Art Of Haptic Technology. Journal of Emerging Technologies and Innovative Research (JETIR).
[10] M. Okamura, A., 2009. Haptic Feedback In Robot-Assisted Minimally Invasive Surgery. [ebook] pp.Volume 19 - Issue 1 - p 102-107. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2701448/.
[11] M. Okamura, A., 2005. Methods For Haptic Feedback In Teleoperated Robot-Assisted Surgery. [ebook] pp.499–508. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1317565/>.
[12] Haptic gloves for virtual reality and robotics | HaptX. (2021). Retrieved 9 June 2021, from https://haptx.com/
[13] Ashrafian, H., Clancy, O., Grover, V. and Darzi, A., 2017. The Evolution Of Robotic Surgery: Surgical And Anaesthetic Aspects. [ebook] London, UK: British Journal of Anaesthesia, pp.72-84. Available at: <https://www.sciencedirect.com/science/article/pii/S0007091217541173>.
[14] S Dai, J. and Kuo, C., 2009. Robotics For Minimally Invasive Surgery: A Historical Review From The Perspective Of Kinematics. [ebook] London, UK, pp.337-354. Available at: <https://www.researchgate.net/publication/226720718_Robotics_for_Minimally_Invasive_Surgery_A_Historical_Review_from_the_Perspective_of_Kinematics>.
[15] Robsurgical.com. n.d. Rob Surgical | Universalising High-Precision Surgery. [online] Available at: <https://www.robsurgical.com/> [Accessed 1 June 2020].
[16] T. Hillel, A., Kapoor, A. and Simaan, N., 2008. Applications Of Robotics For Laryngeal Surgery. [ebook] Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4096143/> [Accessed 3 June 2020].

[17] Clearpath Robotics. n.d. UR5. [online] Available at: <https://store.clearpathrobotics.com/products/universal-robots ur5#:~:text=Overview,payload%20of%20up%20to%205kg.> [Accessed 4 June 2020].
[18] IEEE Spectrum: Technology, Engineering, and Science News. n.d. Full Page Reload. [online] Available at: <https://spectrum.ieee.org/automaton/robotics/industrial-robots/universal-robots-ur3-robotic-arm> [Accessed 4 June 2020].
[19] En.wikipedia.org. n.d. Robotic Arm. [online] Available at: <https://en.wikipedia.org/wiki/Robotic_arm#:~:text=A%20robotic%20arm%20(not%20robotic,of%20a%20more%20complex%20robot.> [Accessed 4 June 2020].
[20] 3dsystems.com. n.d. 3D Printers, Software, Manufacturing & Digital Healthcare | 3D Systems. [online] Available at: <https://www.3dsystems.com/> [Accessed 5 June 2020].
[21] Forcedimension.com. n.d. Force Dimension - Home. [online] Available at: <https://www.forcedimension.com/> [Accessed 5 June 2020].
[22] Servomotor - Wikipedia. Retrieved 6 June 2021, from https://en.wikipedia.org/wiki/Servomotor
[23] Yscadar Cos, A. M (2019). DaVinci robot at Hospital Clinic. Haptic technology to tactile perception in surgical process. Final Biomedical Engineering Degree Project. Universitat de Barcelona.
[24] ROBOTIS e-Manual. n.d. ROBOTIS E-Manual. [online] Available at: <https://emanual.robotis.com/docs/en/parts/controller/opencm904/> [Accessed 5 June 2020].
[25] Robocontroller, A., n.d. Arbotix-M Robocontroller. [online] Trossenrobotics.com. Available at: <https://www.trossenrobotics.com/p/arbotix-robot-controller.aspx> [Accessed 5 June 2020].
[26] Ni.com. 2020. Seleccione Su Edición De Labview - National Instruments. [online] Available at: <https://www.ni.com/es-es/shop/labview/select-edition.html> [Accessed 5 June 2020].
[27] Mathworks.com. 2020. Pricing And Licensing. [online] Available at: <https://www.mathworks.com/pricing-licensing.html> [Accessed 5 June 2020].
[28] En.wikipedia.org. n.d. High-Level Programming Language. [online] Available at: <https://en.wikipedia.org/wiki/High-level_programming_language> [Accessed 5 June 2020].
[29] Retrieved 25 May 2021, from https://all3dp.com/1/best-free-3d-modeling-software-3d-cad-3d-design-software/#solidworks
[30] Top 10 Free PCB Design Software for 2019 - Electronics-Lab.com. (2019). Retrieved 7 June 2021, from https://www.electronics-lab.com/top-10-free-pcb-design-software-2019/
[31] What are the advantages and disadvantages (if any) of Robot Operating System (ROS)? - Quora. Retrieved 31 May 2021, from https://www.quora.com/What-are-the-advantages-and-disadvantages-if-any-of-Robot-Operating-System-ROS
[32] Iso.org. 2020. ISO - 11.040.01 - Medical Equipment In General. [online] Available at: <https://www.iso.org/ics/11.040.01/x/> [Accessed 6 June 2020].
[33] Retrieved 30 May 2021, from https://www.bsigroup.com/globalassets/meddev/localfiles/fr-fr/whitepapers/risk_management_web.pdf
[34] UNE-EN IEC 60601-2-31:2020 (Ratificada) Equipos electromédicos... (2020). Retrieved 30 May 2021, from https://www.une.org/encuentra-tu-norma/busca-tu-norma/norma?c=N0063625
[35] UNE-EN 62366-1:2015/A1:2020 (Ratificada) Productos sanitarios.... (2020). Retrieved 30 May 2021, from https://www.une.org/encuentra-tu-norma/busca-tu-norma/norma/?c=N0064517

[36] Irwin, L. An introduction to ISO 27701: the international standard for data privacy - IT Governance Blog En. Retrieved 30 May 2021, from https://www.itgovernance.eu/blog/en/iso-27701-the-new-international-standard-for-data-privacy
[37] Retrieved 30 May 2021, from https://www.bsigroup.com/globalassets/meddev/localfiles/it-it/webinars/bsi-md-symbols-and-information-to-be-provided-by-the-manufacturer-webinar.pdf

12. Annex

12.1. Inertial Mass Unit sensor implementation

```
#include "src/RoboticsUB.h"

IMU imu;

float * rpy; // Pointer to read RPY
char instruction = 0; // For incoming serial data

void setup()
{

  Serial.begin(115200);

  imu.Install();

  attachInterrupt(digitalPinToInterrupt(8), imuISR, RISING);

}

void loop()
{

  if (Serial.available() > 0) {

    // read the incoming byte:
    instruction = Serial.read();

    switch(instruction){
      case 'A':

        rpy = imu.GetRPY();

        Serial.println(String(rpy[0], 4));
        Serial.println(String(rpy[1], 4));
        Serial.println(String(rpy[2], 4));

        break;

      default:
        break;
    }

    instruction = NULL;

  }
}

void imuISR(void) {

  imu.ReadSensor();


}
```

12.2. Four servo motors torque and movement with IMU sensor

```
#include "src/RoboticsUB.h"
#include <Servo.h>

IMU imu;
Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;


int PIN_IMU_VCC = 4;
int PIN_IMU_INT = 5;
float *rpw;             // Pointer to read RPW
float *q;               // Pointer to quaternion
char instruction = 0; // For incoming serial data

int Pin_R1 = A0;        // Analogic pin used by R1 (Servo1)
int Pin_R2 = A1;        // Analogic pin used by R1 (Servo1)
int Pin_R3 = A2;        // Analogic pin used by R1 (Servo1)
int Pin_R4 = A3;

float R = 1.6;        // Resistance value
float torque1 = 0;      // Indicated as current (ampere)
float torque_int1 = 0;
float torque2 = 0;      // Indicated as current (ampere)
float torque_int2 = 0;
float torque3 = 0;      // Indicated as current (ampere)
float torque_int3 = 0;
float torque4 = 0;      // Indicated as current (ampere)
float torque_int4 = 0;

const unsigned long period_milis = 200; //Time for torque output
unsigned long current_milis1 = 0;
unsigned long previous_milis1 = 0;

unsigned long current_milis2 = 0;
unsigned long previous_milis2 = 0;

unsigned long current_milis3 = 0;
unsigned long previous_milis3 = 0;

unsigned long current_milis4 = 0;
unsigned long previous_milis4 = 0;


float motor_angle_X = 0;  // Motor angle
float motor_angle_Y = 0;  // Motor angle
float motor_angle_Z = 0;  // Motor angle

void setup()
{

  Serial.begin(115200);

  // Power the IMU from pin to reset
  pinMode(PIN_IMU_VCC, OUTPUT);
  digitalWrite(PIN_IMU_VCC, LOW);
  delay(100);
  digitalWrite(PIN_IMU_VCC, HIGH);
  delay(100);
  imu.Install();
  servo1.attach(9);
  servo2.attach(10);
  servo3.attach(11);
  servo4.attach(6);
```

```


}

void loop()
{

  current_milis1 = millis();
  current_milis2 = millis();
  current_milis3 = millis();
  current_milis4 = millis();

  if (digitalRead(PIN_IMU_INT) == HIGH) {
    imu.ReadSensor();
    rpw = imu.GetRPW();
    q = imu.GetQuaternion();
  }

  // Angle range from 0 to 180 degrees
  if (rpw[0] <= 180 && rpw[0] >= 0)
  {
    motor_angle_X = rpw[0];
  }

  servo2.write(motor_angle_X);

  if (current_milis1-previous_milis1>=period_milis){
    torque1=torque_int1;
    torque_int1=0;
    previous_milis1=current_milis1;
    }

    else{
      torque_int1 += analogRead(Pin_R1) * (3.3 / 1023.0) / R;
        }


      // Angle range from 0 to 180 degrees
      if (rpw[1] <= 180 && rpw[1] >= 0)
      {
        motor_angle_Y = rpw[1];
      }
      servo3.write(motor_angle_Y);

      if (current_milis2-previous_milis2>=period_milis){
        torque2=torque_int2;
        torque_int2=0;
        previous_milis2=current_milis2;
        }

        else{
          torque_int2 += analogRead(Pin_R2) * (5 / 1023.0) / R;
        }


      // Angle range from 0 to 180 degrees
      if (rpw[2] <= 180 && rpw[2] >= 0)
      {
        motor_angle_Z = rpw[2];
      }

      // float angle_yaw_2 = - motor_angle_Z;
      servo1.write(motor_angle_Z);
      servo4.write(motor_angle_Z);

      //Serial.println(angle_yaw_2);

      if (current_milis3-previous_milis3>=period_milis){
        torque3=torque_int3;
        torque_int3=0;
        previous_milis3=current_milis3;
        }

        else{
          torque_int3 += analogRead(Pin_R3) * (3.3 / 1023.0) / R;
        }

      if (current_milis4-previous_milis4>=period_milis){
        torque4=torque_int4;
        torque_int4=0;
        previous_milis4=current_milis4;
        }

        else{
          torque_int4 += analogRead(Pin_R4) * (3.3 / 1023.0) / R;
        }

        if (Serial.available() > 0){
          instruction = Serial.read();

        switch (instruction)
        {
        case 'A':

          Serial.println(String(rpw[0], 4));
          Serial.println(String(rpw[1], 4));
          Serial.println(String(rpw[2], 4));
          //Serial.println(String(torque, 4));

          break;

        case 'B':

          Serial.println(String(q[0], 4));
          Serial.println(String(q[1], 4));
          Serial.println(String(q[2], 4));
          Serial.println(String(q[3], 4));
          //Serial.println(String(torque, 4));

          break;

        default:
          break;
        }

        instruction = NULL;

      }
      Serial.println(torque2); //only for test in arduino. Comment it to go to roboDK!
      }
```

12.3. Integration of buzzer and vibration motor and a torque from RoboDK

```
const int buzzer = D3; //buzzer to arduino pin 3
const int vibr = D5;
bool condicio = HIGH;
int torque_;
int thres = 100;
int valor = torque_ - thres;


void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  Serial.setTimeout(1);
  pinMode(vibr, OUTPUT); // Set buzzer - pin 3 as an output
  pinMode(buzzer, OUTPUT); // Set buzzer - pin 3 as an output
}

void loop() {
  // put your main code here, to run repeatedly:
  while (!Serial.available());
  torque_ = Serial.readString().toInt();
  buzzer_function();
  vibration_motor();
}
```

12.4. Buzzer and Vibration motor depending of the torque threshold

```
void buzzer_function(void){
    if (torque_ > thres){
      tone(buzzer, 2*torque_); // Send sound signal...
    } else {
      noTone(buzzer);     // Stop sound...
    }
}

void vibration_motor(void){
    if (torque_ > thres){
      // analogWrite(pin, value) --> value may go from 0 to 255 --> 0 minimum; 255 maximum
      analogWrite(vibr,torque_/2);
    } else {
      analogWrite(vibr,0);
    }
}
```

12.5. Reading torque without IMU sensor

```
#include "src/RoboticsUB.h"
#include <Servo.h>

Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4; // ens falta una R

int Pin_R1 = A0;      // Analogic pin used by R1 (Servo1)
int Pin_R2 = A1;      // Analogic pin used by R2 (Servo2)
int Pin_R3 = A2;      // Analogic pin used by R3 (Servo3)
int Pin_R4 = A3;      // Analogic pin used by R4 (Servo4)

float R = 1.6;        // Resistance value
float torque1 = 0;      // Indicated as current (ampere)
float torque_int1 = 0;
float torque2 = 0;      // Indicated as current (ampere)
float torque_int2 = 0;
float torque3 = 0;      // Indicated as current (ampere)
float torque_int3 = 0;
float torque4 = 0;      // Indicated as current (ampere)
float torque_int4 = 0;

const unsigned long period_milis = 200; //Time for torque output
unsigned long current_milis1 = 0;
unsigned long previous_milis1 = 0;

unsigned long current_milis2 = 0;
unsigned long previous_milis2 = 0;

unsigned long current_milis3 = 0;
unsigned long previous_milis3 = 0;

unsigned long current_milis4 = 0;
unsigned long previous_milis4 = 0;

float motor_angle_X = 0;  // Motor angle
float motor_angle_Y = 0;  // Motor angle
float motor_angle_Z = 0;  // Motor angle

bool condicio = HIGH;

float rpw_1 = 0;
float rpw_2 = 90;
float rpw_3 = 0;
float rpw_4 = 0;


void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  // Serial.setTimeout(1);
  servo1.attach(9);
  servo2.attach(10);
  servo3.attach(11);
  servo4.attach(6);
}

void loop() {
  // put your main code here, to run repeatedly:

  current_milis1 = millis();
  current_milis2 = millis();
  current_milis3 = millis();
  current_milis4 = millis();

  if (rpw_1 <= 180 && rpw_1 >= 0){
    motor_angle_X = rpw_1;
  }

  servo1.write(motor_angle_X);

  if (current_milis1-previous_milis1>=period_milis){
    torque1=torque_int1;
    torque_int1=0;
    previous_milis1=current_milis1;
    }

    else{
      torque_int1 += analogRead(Pin_R1) * (3.3 / 1023.0) / R;
    }

  if (rpw_2 <= 180 && rpw_2 >= 0)
  {
    motor_angle_Y = rpw_2;
  }
  servo2.write(motor_angle_Y);


  if (current_milis2-previous_milis2>=period_milis){
    torque2=torque_int2;
    torque_int2=0;
    previous_milis2=current_milis2;
    }

    else{
      torque_int2 += analogRead(Pin_R2) * (5 / 1023.0) / R;
    }

      if (rpw_3 <= 180 && rpw_3 >= 0)
      {
        motor_angle_Z = rpw_3;
      }
      servo3.write(motor_angle_Z);
      servo4.write(motor_angle_Z);

      if (current_milis3-previous_milis3>=period_milis){
        torque3=torque_int3;
        torque_int3=0;
        previous_milis3=current_milis3;
        }

        else{
          torque_int3 += analogRead(Pin_R3) * (3.3 / 1023.0) / R;
        }


      if (current_milis4-previous_milis4>=period_milis){
        torque4=torque_int4;
        torque_int4=0;
        previous_milis4=current_milis4;
        }

        else{
          torque_int4 += analogRead(Pin_R4) * (3.3 / 1023.0) / R;
        }

      if (Serial.available()>0){
        Serial.println(String(torque1,4));
        Serial.println(String(torque2,4));
        Serial.println(String(torque3,4));
        Serial.println(String(torque4,4));
      }
```

```
        //Serial.print(" Torque 2= ");
        Serial.println(torque2);
        //Serial.print(" Torque 3= ");
        //Serial.print(torque3);
        //Serial.println();
    }
```

12.6. IMU simulation in RoboDK

```python
import serial
import time
import math
import tkinter as tk


# RoboDK API: import the robolink library (bridge with RoboDK)
from robolink import *
# Robot toolbox: import the robodk library (robotics toolbox)
from robodk import *

# Variables definition
# TCP end-effector respect the Flange
X=0
Y=-60
Z=320


# Lets bring some time to the system to stablish the connetction
time.sleep(2)

# Establish a link with the simulator
RDK = Robolink()

# -------------------------------------------------------------------------------
# Simulator setup
# -------------------------------------------------------------------------------

# Retrieve all items (object in the robodk tree)
# Define the "robot" variable with our robot (UR5e)
robot = RDK.Item ('UR5e')

# Define the "tcp" variable with the TCP of Endowrist needle
tcp_tool = RDK.Item('TCP_Endowrist')

# Performs a quick check to validate items defined
if robot.Valid():
    print('Robot selected: ' + robot.Name())
if tcp_tool.Valid():
    print('Tool selected: ' + tcp_tool.Name())

# Robot Flange with respect to UR5e base Frame
print ('Robot POSE is: ' + repr(robot.Pose()))
# Tool frame with respect to Robot Flange
print ('Robot POSE is: ' + repr(robot.PoseTool()))
# Tool frame with respect to Tool frame
print ('TCP pose is: ' + repr(tcp_tool.Pose()))

# -------------------------------------------------------------------------------
#  Establish the connection on a specific port (COM5)
arduino = serial.Serial("COM7", 115200, timeout=1)

window = tk.Tk()
window.geometry('200x200')
etiqueta = tk.Label(window, text = "")
```

```python
etiqueta.pack()

def roll_pitch_yaw():
    global etiqueta
    try:
        while True:

            # Requesting data to Arduino (command A)
            arduino.write(b'A')

            # Storing received data
            roll_str = arduino.readline().strip()
            # roll_str = str(90)
            pitch_str = arduino.readline().strip()
            # pitch_str = str(180)
            yaw_str = arduino.readline().strip()
            # yaw_str = str(0)
            # torque_str = arduino.readline().strip()
            # torque_str = str(90)

            print(roll_str, pitch_str, yaw_str)

            # Convert variable values from string to float
            roll = float(roll_str)
            pitch = float(pitch_str)
            yaw = float(yaw_str)
            # torque = float(torque_str)

            # Convert from degrees to radians R,P,Y angles
            R = math.radians(roll)
            P = math.radians(pitch)
            W = math.radians(yaw)

            # Calculate the POSE matrix (UR)
            pose_matrix = transl([X, Y, Z])*rotx(pi)*rotx(-R)*roty(-P)*rotz(-W)
            print ('The POSE matrix with RPY is: ' + repr(pose_matrix))
            tcp_tool_pose = tcp_tool.setPoseTool(pose_matrix)

            var_1 = ('r = ',R, '\n', 'p = ',P,'\n','y = ',W,'\n')
            etiqueta.config(text = var_1)
            window.update()


    except KeyboardInterrupt:
            print("Communication stopped.")
            pass



    choice_1 = tk.Button(window, text = 'RPY', command=roll_pitch_yaw)

    choice_1.pack()

    roll_pitch_yaw()

    # ------------------------------------------------------------------------------
    # Disconnect Arduino
    # ------------------------------------------------------------------------------
    print("Disconnecting Arduino...")
    arduino.close()
```

12.7. Torque value from RoboDK sent to Arduino buzzer and vibration motor

```python
from robolink import *    # RoboDK API
from robodk import *      # Robot toolbox
import serial
import time
import math
import tkinter as tk
RDK = Robolink()

robot = RDK.Item ('UR5e')
tcp_tool = RDK.Item('TCP_Endowrist')

# Performs a quick check to validate items defined
if robot.Valid():
    print('Robot selected: ' + robot.Name())
if tcp_tool.Valid():
    print('Tool selected: ' + tcp_tool.Name())

# Robot Flange with respect to UR5e base Frame
print ('Robot POSE is: ' + repr(robot.Pose()))
# Tool frame with respect to Robot Flange
print ('Robot POSE is: ' + repr(robot.PoseTool()))
# Tool frame with respect to Tool frame
print ('TCP pose is: ' + repr(tcp_tool.Pose()))

arduino = serial.Serial("COM7", 115200, timeout=1)

def write_torque_to_arduino(torque_):
    arduino.write(bytes(torque_, 'utf-8'))
    time.sleep(0.05)
    T = arduino.readline()
    return T

while True:
    torq = str(input("Type torque: "))
    torque_value = write_torque_to_arduino(torq)

# Disconnect Arduino
# --------------------------------------------------------------------
-----------
print("Disconnecting Arduino...")
arduino.close()
```