POLITECNICO DI MILANO

Corso di Laurea Specialistica in Ingegneria Biomedica

Facoltà di Ingegneria dei Sistemi

Dipartimento di Bioingegneria

Tesi di Laurea Specialistica

SINGULARITY-ROBUST KINEMATIC CONTROL OF A NOTES ROBOT

Relatori: Prof. Pietro CERVERI

Prof. Tarcisio Antonio HESS-COELHO

Autore:

Daniel José DAMAS FOLLADOR

Matricola 734847

Anno Accademico 2011-2012

Acknoweldgments

To Aline, for her patience and unconditional support

To my professors Pietro and Tarcisio, for their attention and availability

To my parents, Maria José e Wilson, for their love, comprehension and for making possible every one of my accomplishments

Abstract

Driven by the surgeon's desire for less-invasive procedures, a new technique called NOTES (Natural Orifice Transluminal Endoscopic Surgery) has emerged. The present instrumentation used for this type of surgery is adapted from Endoscopy and is not perfectly adequate for many reasons: lack of dexterity, lack of tactile feedback, fewer degrees of freedom than desired. Those problems were the motivation for a Italian research group to develop a robotic manipulator with several advantages over the current technology. The snake-like mechanical design of the robot contains three kinematic singularities whose physical interpretation were one of the objectives of this work. The second main goal of this work was to design an adequate kinematic control algorithm for avoiding singularities by using geometric redundancy from other degrees of freedom. For that matter, four different trajectories were tested with four different algorithms and the results were evaluated in terms of tracking error in the Cartesian space and in the joint space.

Sommario

Spinto dal desiderio del chirurgo per meno invasivi, una nuova tecnica chiamata NOTES (Natural Orifice Transluminal Endoscopic Surgery) è emersa. La strumentazione attualmente utilizzata per questo tipo di chirurgia è adattato da endoscopia e non è perfettamente adeguata per molte ragioni: mancanza di destrezza, la mancanza di feedback tattile, un minor numero di gradi di libertà di quanto desiderato. Questi problemi sono stati la motivazione per un gruppo di ricerca italiano per sviluppare un manipolatore robotico con diversi vantaggi rispetto alla tecnologia attuale. Il serpente-come il disegno meccanico del robot contiene tre singolarità cinematica cui interpretazione fisica sono stati uno degli obiettivi di questo lavoro. Il secondo obiettivo principale di questo lavoro è stato quello di progettare un algoritmo adeguato controllo cinematico per evitare la singolarità utilizzando la ridondanza geometrica da altri gradi di libertà. Per questo, quattro diverse traiettorie sono stati testati con quattro diversi algoritmi ei risultati sono stati valutati in termini di *tracking error* nello spazio cartesiano e nello spazio dei giunti.

Contents

Figure Index

1. Introduction

The earliest evidence of a surgical procedure occurred early in the pre-history. The registration of a trepanation (opening of one or more holes in the skull to relieve intracranial pressure) was found in cave paintings and later in historical records (Capasso, 2002). Since then, surgery has evolved throughout history, new techniques and instruments have been developed and much has been learned about the anatomy, physiology, risk of infection and contamination, culminating today about 20 million Americans operated yearly (Roan, 2005).

We classify an operation according to some criteria: (American College of Surgeons, 2007)

- Based on **urgency**, according to the risk to patient survival, surgery is considered elective, urgent or optional;
- Based on the **purpose**, an exploratory surgery may be (to confirm diagnosis) or therapy;
- Based on the **type of procedure**, the surgery may vary according to the standard intervention (e.g., amputation, plastic surgery, transplantation, etc.);
- Based on **anatomical site**, taking as examples the cardiovascular surgery, orthopedic and gastrointestinal.
- By type of **equipment** used, may involve classical tools (such as a scalpel and forceps), more modern tools such as laser or even mechatronic instruments, such as *Intuitive Surgical's Da Vinci Robot* (Intuitive Surgical, 2010)*;*
- By degree of **invasiveness**, surgery may be:
    - open, which will involve the opening of several tissues of the surgeon to achieve the desired target;
    - minimally invasive surgery, which are made possible through the lower courts, such as in laparoscopic surgery, angioplasty and surgery NOTES

1.1. Minimally Invasive Surgery

A minimally invasive surgery is a procedure that involves a different set of tools for the same objective of the open surgery, but with far less damage to biological tissues. It is important to point out that all the instrumentation for that type of surgery is specially designed and submitted to constant innovation.

By comparison, minimally invasive surgeries have some advantages over open surgery. A study in the department of surgery at Henry Ford Hospital in Detroit in the USA (Velanovich, 2000) showed that in addition to reducing pain for patients and quicker return to normal function, the minimally invasive surgery (MIS) also result in better quality of life compared with open techniques.

1.2. Laparoscopic Surgery

Laparoscopic or "minimally invasive" surgery is a specialized technique for performing surgery. In the past, this technique was commonly used for gynecologic surgery and for gall bladder surgery. Over the last 10 years the use of this technique has expanded into intestinal surgery.

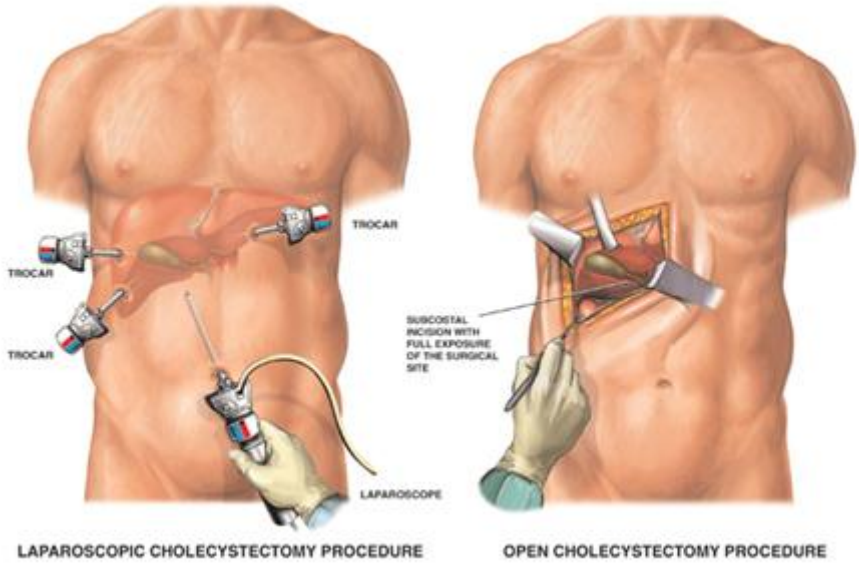

Figure 1 - Laparoscopy vs. open surgery

As shown in Figure 1, in traditional "open" surgery the surgeon uses a single incision to enter into the abdomen. Laparoscopic surgery uses several 0.5-1cm incisions. Each incision is called a "port." At each port a tubular instrument known as a trochar is inserted. Specialized instruments and a special camera known as a laparoscope are passed through the trochars during the procedure. At the beginning of the procedure, the abdomen is inflated with carbon dioxide gas to provide a working and viewing space for the surgeon. The laparoscope transmits images from the abdominal cavity to high-resolution video monitors in the operating room. During the operation the surgeon watches detailed images of the abdomen on the monitor. This system allows the surgeon to perform the same operations as traditional surgery but with smaller incisions.

In certain situations a surgeon may choose to use a special type of port that is large enough to insert a hand. When a hand port is used the surgical technique is called "hand assisted" laparoscopy. The incision required for the hand port is larger than the other laparoscopic incisions, but is usually smaller than the incision required for traditional surgery. Compared to traditional open surgery, patients often experience less pain, a shorter recovery, and less scarring with laparoscopic surgery. (American Society of Colon & Rectal Surgeons, 2008)

1.3. Robotic Surgery

Robotic surgery is a technique in which a surgeon performs surgery using a computer that remotely controls very small instruments attached to a robot.

This procedure is done under general anesthesia. The surgeon sits at a computer station nearby and directs the movements of a robot. Small instruments are attached to the robot's arms. The surgeon first inserts these instruments into the patient's body through small surgical cuts. Under the surgeon's direction, the robot matches the

doctor's hand movements to perform the procedure using the tiny instruments, as shown in Figure 2.

Figure 2 - Example of robotic surgery system

A thin tube with a camera attached to the end of the endoscope allows the surgeon to view highly magnified three-dimensional images on a monitor in real time. Robotic surgery is a type of procedure that is similar to laparoscopic surgery. It also can be performed through smaller surgical cuts than traditional open surgery. The small, precise movements that are possible with this type of surgery give it some advantages over standard endoscopic techniques (Oleynikov, 2008).

Sometimes robotic-assisted laparoscopy can allow a surgeon to perform a less-invasive procedure that was once only possible with more invasive open surgery. Once it is placed in the abdomen, a robotic arm is easier for the surgeon to use than the instruments in endoscopic surgery. The robot reduces the surgeon's movements (for example, moving 1/2 inch for every 1 inch the surgeon moves), which reduces some of the hand tremors and movements that might otherwise make the surgery less precise. Also, robotic instruments can access hard-to-reach areas of your body more easily through smaller surgical cuts compared to traditional open and laparoscopic surgery.

During robotic surgery, the surgeon can more easily see the area being operated on. The surgeon is also in a much more comfortable position and can move in a more natural way than during endoscopy. Robotic surgery may be used for a number of different procedures, including (Oleynikov, 2008):

- Coronary artery bypass
- Gallbladder removal
- Hip replacement
- Hysterectomy
- Kidney removal
- Kidney transplant
- Mitral valve repair
- Radical prostatectomy
- Tubal ligation

Robotic surgery cannot be used for some complex procedures. For example, it is not appropriate for certain types of heart surgery that require greater ability to move instruments in the patient's chest, with the present technology.

1.4. NOTES

NOTES (Natural Orifice Transluminal Endoscopic Surgery) was driven by the surgeon’s desire for less invasive procedures and more minimal access, the technique is based on the concept that the peritoneal cavity can be accessed through natural orifices. NOTES designates a surgical procedure that utilizes one or more patent natural orifice of the body with the intention to puncture a hollow viscera in order to enter an otherwise inaccessible body cavity. (Kalloo, Singh, Jagannath, Niiyama, Hill, & Vaughn, 2004)

Natural orifices usually include the mouth (reaching the stomach), anus (reaching the colon), vagina (reaching the uterus) and urethra (reaching the bladder). Theoretically the advantages of a NOTES procedure over the laparoscopic approach comes from avoiding an

external abdominal incision, a less invasive procedure that allows minimization of anesthesia and analgesia and a reduction in postoperative abdominal wall pain, wound infection, hernia formation and adhesions. (Bowman, 2006)

Due to its novel concept and the ongoing medical process, the connotation and classification of NOTES are still not definite and are sometimes controversial. According to the present status of NOTES studies, NOTES procedures are mainly divided into two categories; "pure" NOTES and "hybrid" NOTES. (Geoffrey, Timothy, Jeffrey, Mihir, Edward, & Ralph, 2008)

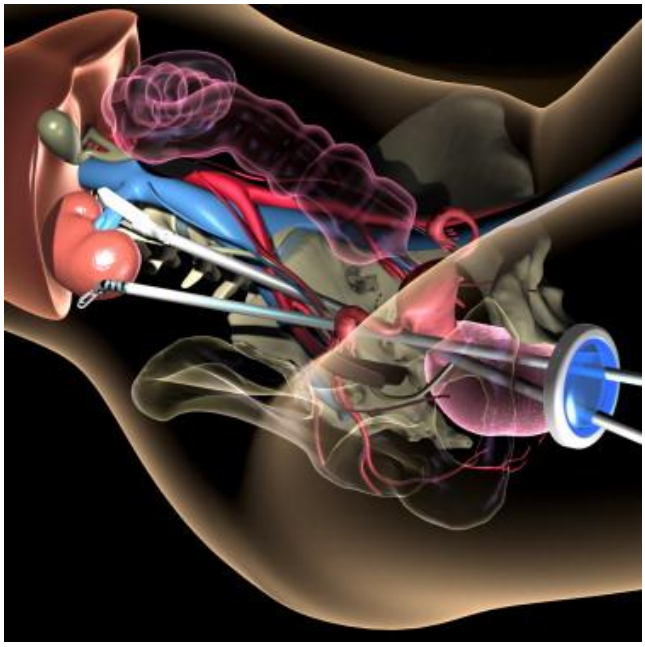

Figure 3 - Pure NOTES transvaginal access

The "pure" NOTES refers to a NOTES procedure that is completed without any transabdominal ports, including the umbilicus, as shown in Figure 3. The "hybrid" NOTES refers to the "mixed" technologies using transabdominal instrumentation to facilitate the NOTES procedure.

For analyzing the proper instrumentation to NOTES, it is valid to look back at some procedures that have been done so far. (Kalloo, Singh, Jagannath, Niiyama, Hill, & Vaughn, 2004) described a cholecystectomy in a porcine model using a series of endoscopic accessories such as the needle-knife, pull-type sphincterotome and

dilation balloons. The transgastric access was difficult in terms of identifications, manipulation and resection of the gallbladder.

(Pai, Fong, Bundga, Odze, Rattner, & Thompson, 2006) in the other hand, executed a transcolonic access because they believed that it would allow a better visualization and endoscope stability because of en face orientation to organs in the upper abdomen.

(Haber, et al., 2008) presented an initial experience of robotic NOTES using the Da Vinci™ surgical system. Ten female pigs were submitted into 10 pyeloplasties, 10 partial nephrectomies and 10 radical nephrectomies successfully using a hybrid approach (umbilical and transvaginal incisions). The intraoperative data showed small operative time and blood loss, but some limitations were perceived: there were 5 episodes of conflicts between instruments (representing 17% of the sample), 3 episodes of unwanted contacts between endoscope and instruments (9%) and in 3 episodes (9%), the instrument could not reach the kidney.

In short, NOTES is a very promising surgery technique with several applications, including cancer surgery (Rieder & Swanstrom, 2011). Many researchers (either medical doctors or biomedical engineers) are looking for technologies and practice to improve this type of surgery and improve patient outcomes.

2. NOTESNAIL Project

2.1. Motivation

Although NOTES is a very promising technique, some studies have been conducted about the quality of manipulability of the instrumentation currently in use and its limitations. One of the major concerns, existent even in the laparoscopic surgery, is the hand-eye coordination that limits three-dimensional triangulation. That issue is accented in NOTES given that camera and instruments are in the same axis, as shown in Figure 4.

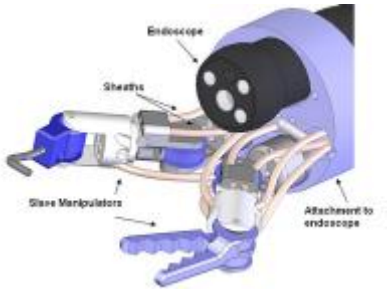

Figure 4 - NOTES instrument proposed in Singapore

Another major obstacle for the surgeon is the flexibility of an endoscope. In one hand, the instrument has to be flexible in order to arrive at the correct site of operation. On the other hand, the instrument has to be rigid in order to perform forces and pressures adequate to the given task. (Bowman, 2006)

(Cerveri, 2008) suggests that NOTES operative instruments cannot be simply adapted from other endoscopic or laparoscopic surgery because of its significantly different working conditions:

- the surgeon cannot rotate the external portion of the instrument in order to orient the tool inside the patient's body;
- the first part of the instrument (proximal part), which does not enter the peritoneal cavity, must reach the entrance incision which may be quite far from the external opening of the natural orifice;
- the second part of the instrument (operative part, entering the peritoneal cavity) needs several degrees of freedom to reach

the operative region, more than in endoscopic instruments where 4 degrees of freedom are obtained just by rotating the external portion of the instrument outside the patient's body;

- mechanical transmission of motion to the operative part of the instrument from a handle outside the orifice is not feasible due to length and geometry of the proximal part of the instrument.
- The loss of force feedback also represents an issue. It is related to minimally invasive surgery in general and it is particularly relevant with NOTES because the instrument is longer with the handle significantly far from the operative region.

That need to overcome these limitations suggest the benefits of specific surgical instruments with high dexterity were the motivation for a research team in Italy composed by students and professors of 4 major universities: Università degli studi di Genova, Politecnico di Milano, Università degli studi di Bergamo and Università degli studi de L'Aquila.

The research project has two breakthrough concepts: snail architecture and variable stiffness actuation. Its first major objective is to build a real-scale prototype with two different distal modules: a VS-grasper and a micro-camera. The whole system scheme is shown in Figure 5.

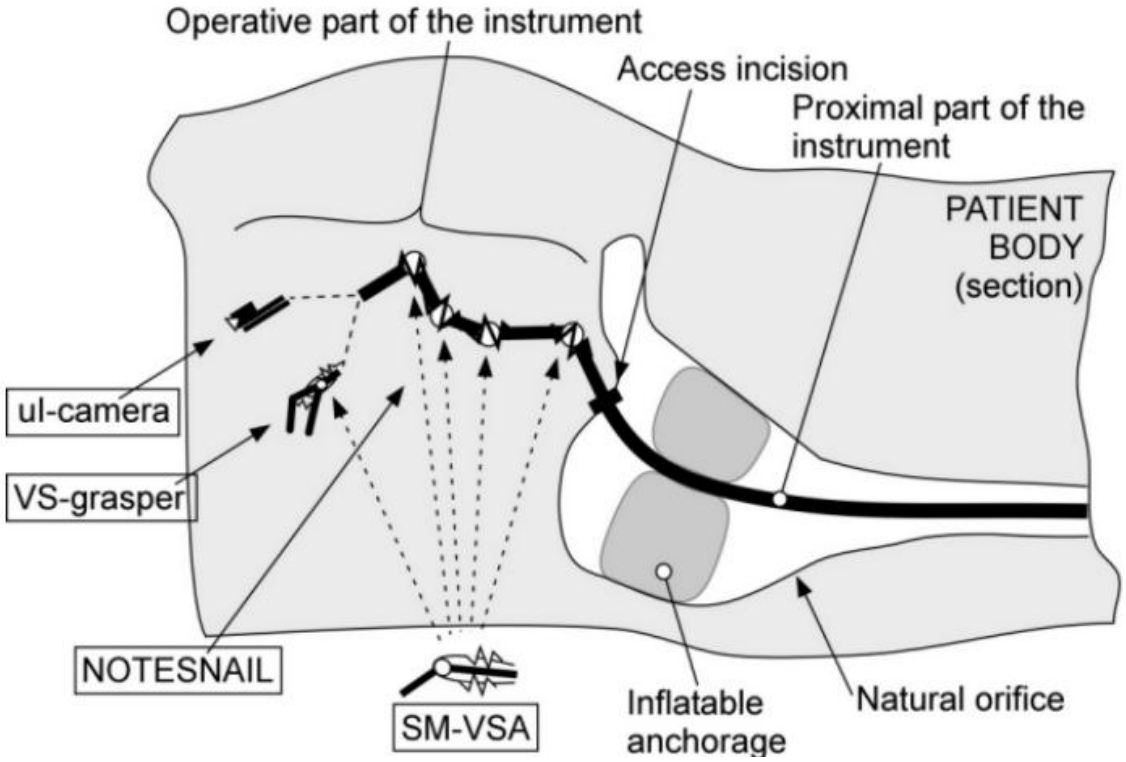

Figure 5 - NOTESnail project scheme

A second major objective is to build a control scheme that allows position and force feedbacks in addition to collision avoidance, what would improve patient and surgeon safety. The last objective is to improve the quality of the visualization of the operating space.
From this point forward, this particular robot will be called NOTESnail.

2.2. Construction details

The NOTESnail mechanical project was done following a series of technical requirements that are common to many surgical applications, plus a series of requirements of our specific use:

- It has to be insertable in a 10-mm-diameter orifice;
- Sufficient maneuverability to move around a target;
- Remotely controllable using a remote console;
- Low heat dissipation and low energy consumption
- Autonomous illumination
- Electromechanical actuation
- Sensor for position encoding and mechanical interaction with surrounding objects

- Sterilizable
- Visual and force servoing control schemes

Previous works done by the Italians have brought a modular concept, with each independent module having two micro-motors moving a bending joint (shown in Figure 6) and a torsion joint (shown in Figure 7). A computer-aided-design representation of the robot's final assemble is shown in Figure 8.

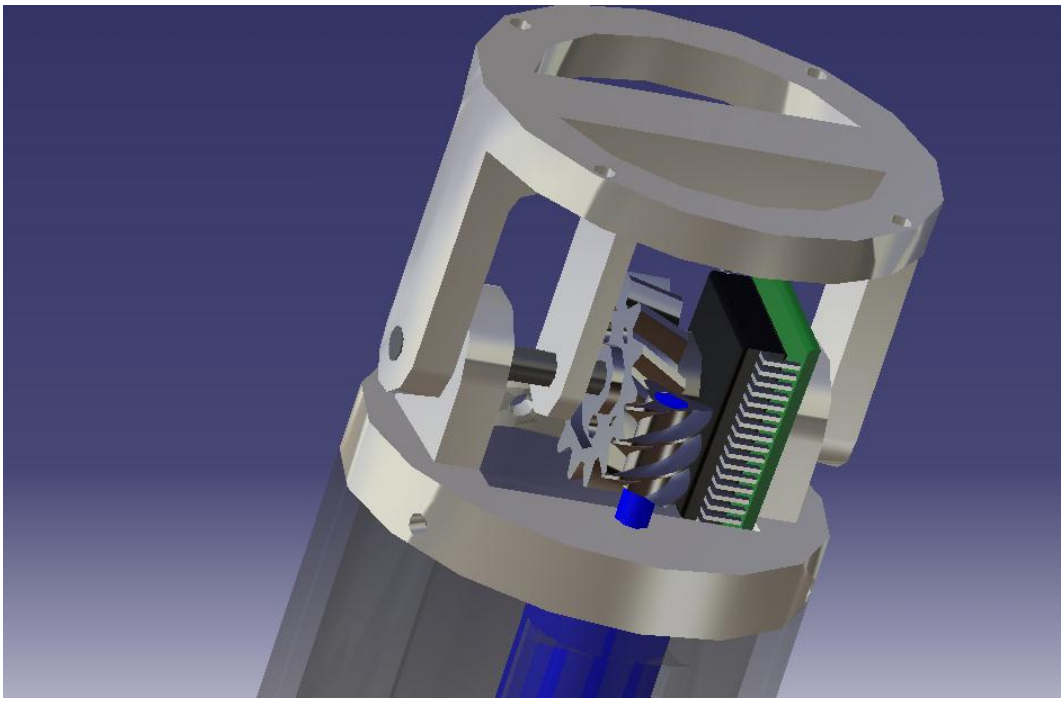

Figure 6 - Bending joint actuator

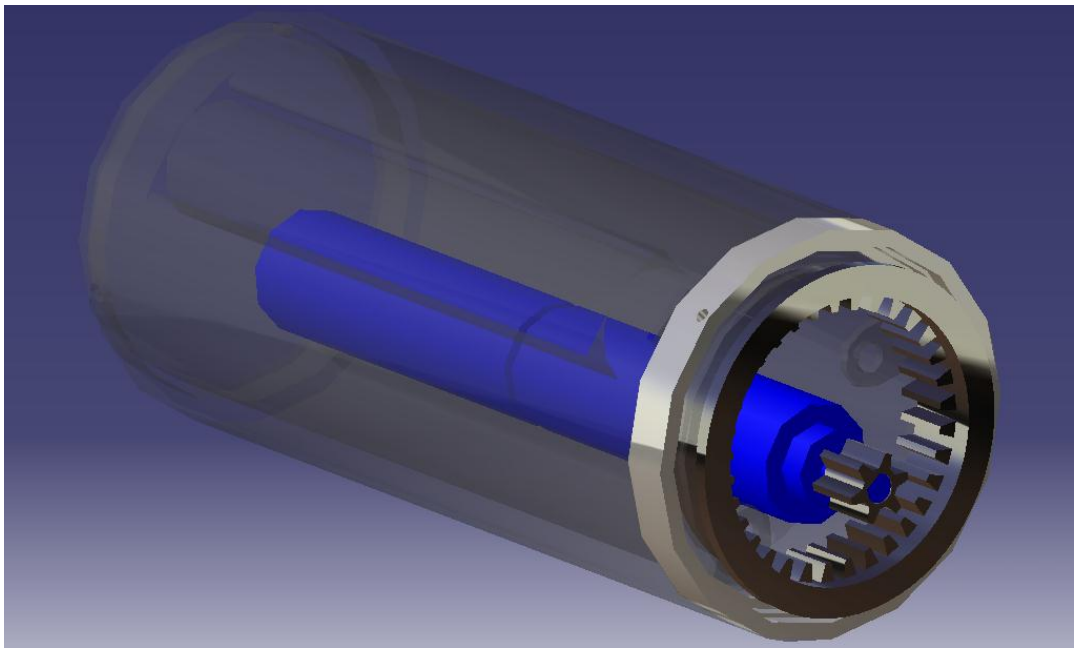

Figure 7 - Torsion joint actuator

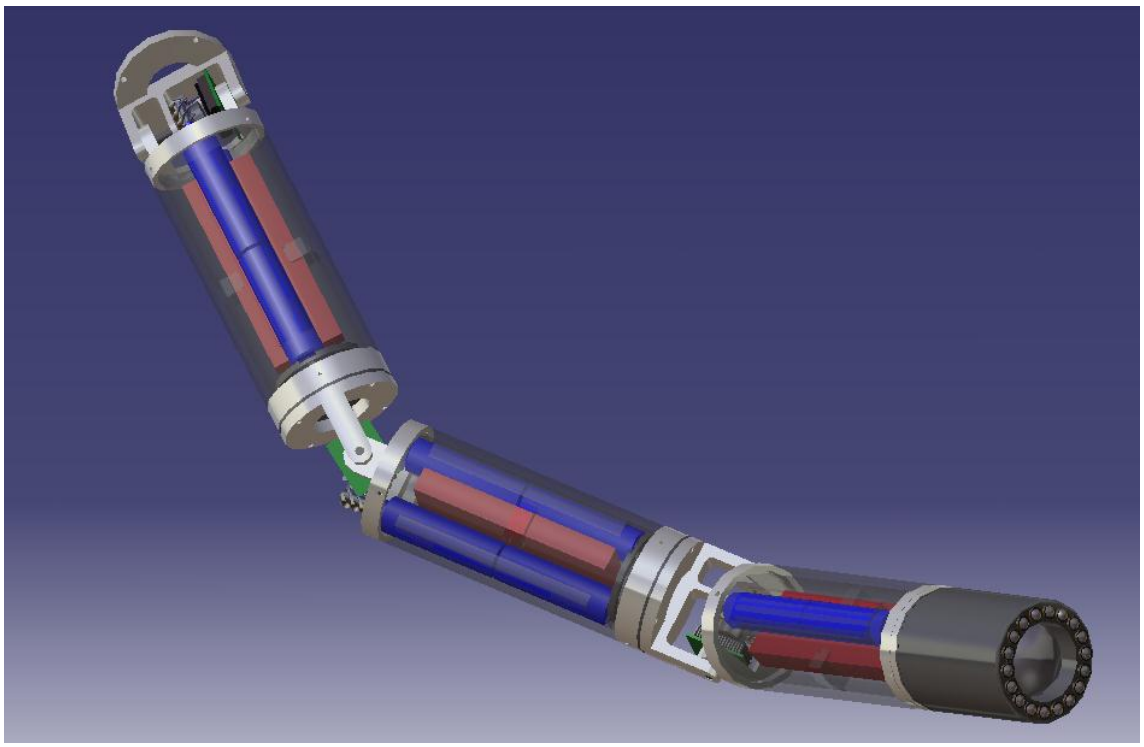

Figure 8 - NOTESnail fully assembled

It is worth noticing the reasons for choosing the number of independent modules. Three modules should be sufficient to provide six degrees of freedom, which will be enough for the surgeon to reach any site of operation with the correct orientation. A larger number of modules would make control algorithms more complicated (what is equivalent to say that it will have a larger computational cost)

Some of the features that are most important for simulating the kinematic and dynamic behavior of NOTESnail are shown in Table 1. Center of mass and moment of inertia were calculated using computer aided design software and it can be found in Appendix A.

Table 1 - NOTESnail physical proprieties

Module length	3.5 cm
Bending joint range	From -0.5 π rad to 0.5 π rad
Torsion joint range	From -0.75 π rad to 0.75 π rad
Module total mass	48 grams
Camera mass	30 grams
Motor inertia	$9.5e^{-10}$ kg/m^2
Viscous friction	$8e^{-7}$ Pa.s

2.3. NOTESnail Forward Kinematics

The forward kinematics problem is concern with the relationship between the individual joints of the robot manipulator and the position and orientation of the tool or end-effector. For computing the forward kinematics equations of NOTESnail, the Danevit-Hartenberg convention was used (Hess-Coelho, 2004):

Table 2 - Denavit-Hartenberg convention parameters

D-H	a	α	d	θ
1	0	-π/2	0	Θ_1
2	0	π/2	L_1	Θ_2
3	0	-π/2	0	Θ_3
4	0	π/2	L_2	Θ_4
5	0	-π/2	0	Θ_5
EE	0	0	L_3	Θ_6

The Denavit-Hartenberg parameters made possible the construction of a homogenous transform matrix (4-by-4 dimension) that provides

position and orientation of the end-effector according to the coordinate system fixed at the base. Using those equations, varying the joint angles along their ranges, it is possible to map the robot's workspace, shown in Figure 9.

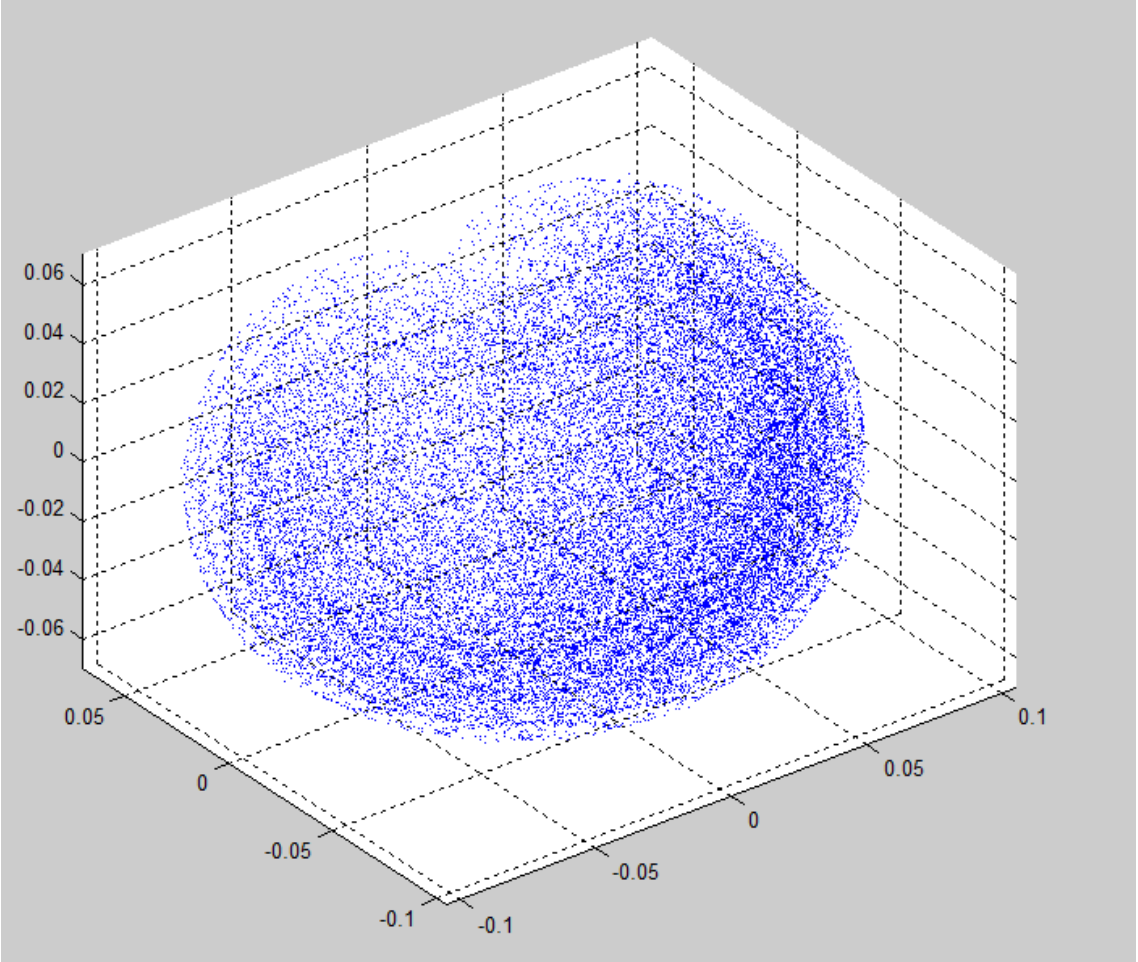

Figure 9- NOTESnail Workspace

3. Singularities study

To better understand the meaning of a robot's singularity, it is necessary to understand the velocity kinematics and the manipulator Jacobian. Mathematically, the forward kinematic equations define a function between the space of Cartesian positions and orientations and the space of joint positions. The velocity relationships are then determined by the Jacobian of this function. (Spong, Hutchinson, & Hutchinson, 2004)

This Jacobian matrix plays an essential rule in the analysis and control of robot motion: from trajectory planning and execution to transformation of forces and torques from the end-effector to the manipulator joints. Another important propriety of a Jacobian matrix (more precisely, its determinant) is the singularity detection.

3.1. Singularity general theory

Given a robot with *n* joints, the $6_x n$ Jacobian J(q) defines a mapping between the vector $\dot{q}$ of joint velocities and the vector $\dot{X}$ of end-effector velocities. Infinitesimally, this defines a linear transformation

$$dX = J(q)dq$$

Equation 1

A set of joint coordinates that causes the Jacobian rank to decrease is called a **singular configuration** or **singularity**. Their importance is given because (Spong, Hutchinson, & Hutchinson, 2004):

- Singularities represent configurations from which certain directions of motion may be impossible.
- At singularities, small end-effector velocities may correspond to very large joint velocities;
- At singularities, small end-effector forces and torques may correspond to very large joint torques.
- Singularities may correspond to boundaries of the manipulator workspace

- Near singularities there will not exist a unique solution to the inverse kinematics problem. In such cases there may be no solution or there may be infinitely many solutions.

(Yoshikawa, 1985) defines a scalar value called "measure of manipulability", which is defined by Equation 2. That measure is given for a certain manipulator at a given configuration. Some authors (Oemoto & Ang Jr., 2007) use that measure for distinguishing different behaviors of their control algorithm.

$$w = \sqrt{\det(J(\theta)J^T(\theta)}$$

Equation 2

Many methods have been proposed to handle singularities and they have been divided by (Oemoto & Ang Jr., 2007) in two main categories. In the first category, a uniform control strategy is adopted throughout the entire workspace, building a continuous function that introduces a slight alteration to the task space specification or its mapping to the manipulator joint space. This generally results in a stable control strategy where the end-effector avoids the singular configuration. The second involves a division of workspace where a different control algorithm is applied to the region around the singularities.

When the manipulator is at a singular configuration, motions and forces along the singular direction are not controllable. If the task includes a motion along that singular direction, it can be achieved using the *null space motion* (which corresponds to minimizing a potential function corresponding to the task goal).

(Oemoto & Ang Jr., 2007) divide singularities into two categories according to the effect that null space motion has on them. Type I singularities (as shown in Figure 10) are those where null space motion creates end- effector motion in the singular direction and causes the end-effector to escape the singular region through this direction. Type II singularities (as shown in Figure 11) are those where null space motion affects only internal joint motion, and

changes the singular directions without affecting the end-effector motion/forces.

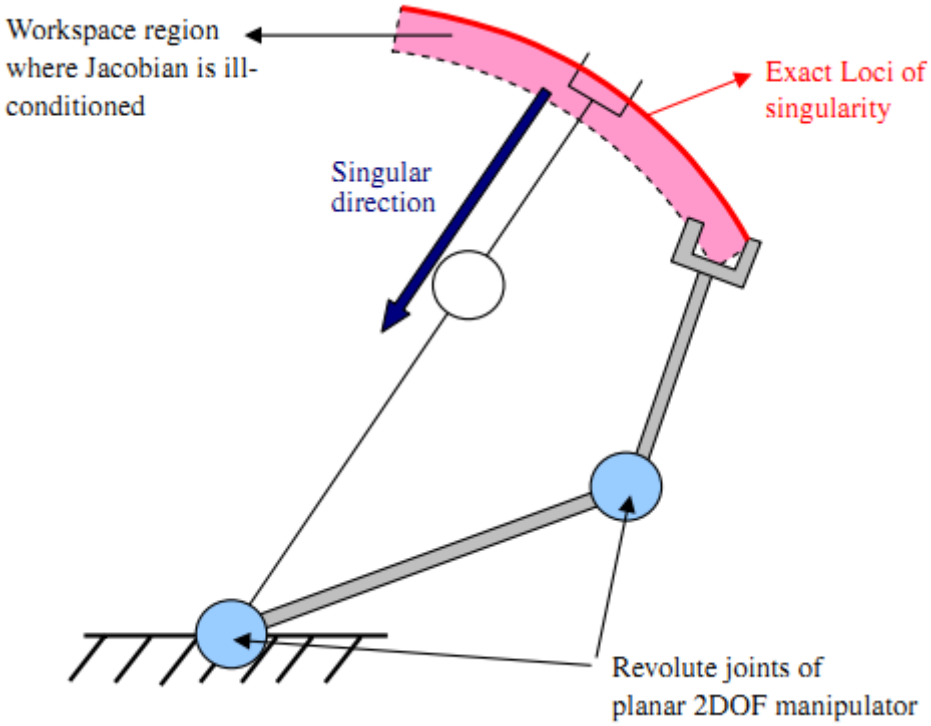

Figure 10 - Example of Type I singularity

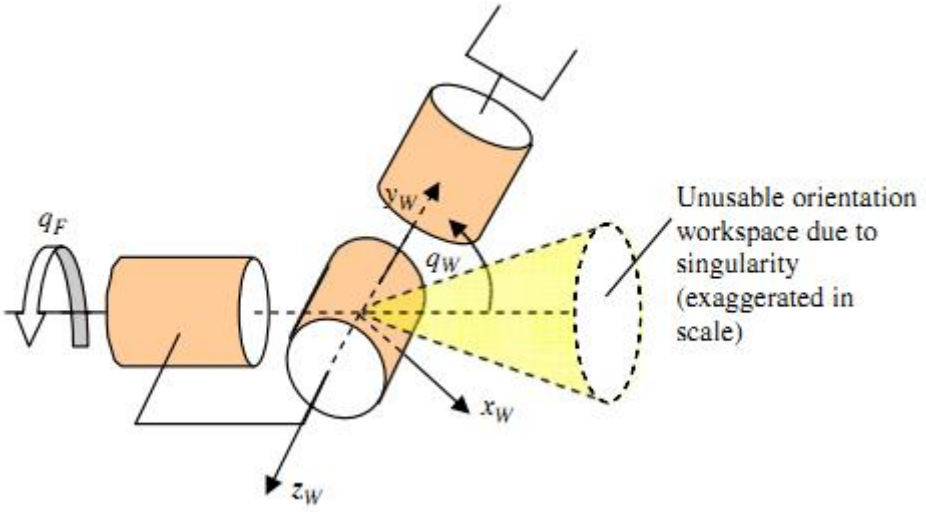

Figure 11 - Example of Type II singularity

(Oemoto & Ang Jr., 2007) have implemented their singularity robust algorithm in the textbooks most famous example Puma560, shown in its rest configuration in Figure 12. Its Jacobian can be partitioned into two parts, corresponding to "arm" and "wrist". Analyzing that matrix

determinant, the authors were able to find three distinct singular configurations corresponding to “elbow lock”, “wrist lock” and “head lock”.

The “elbow lock” is an example of a type I singularity caused by the alignment of two link axis (as show in Figure 13a). Even though q3 is the reducing the measure of manipulability, it’s through its motion that the robot can “fold out” of the singular configuration (as shown in Figure 13b).

The “head lock” is an example of a type II singularity that happens when the wrist point is exactly above the first joint axis (see Figure 14). A null-space motion can rotate the singular direction, allowing the task to be completed without changing the end-effector position or orientation (see Figure 14).

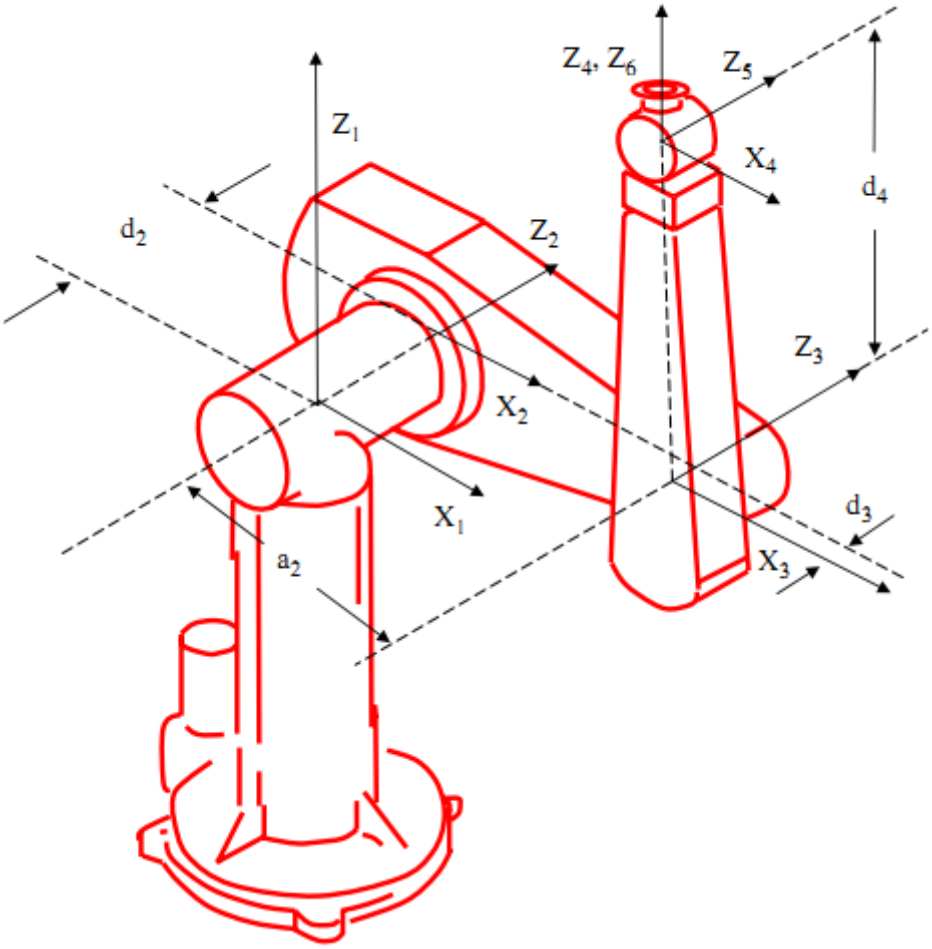

Figure 12 - Puma560 in rest position

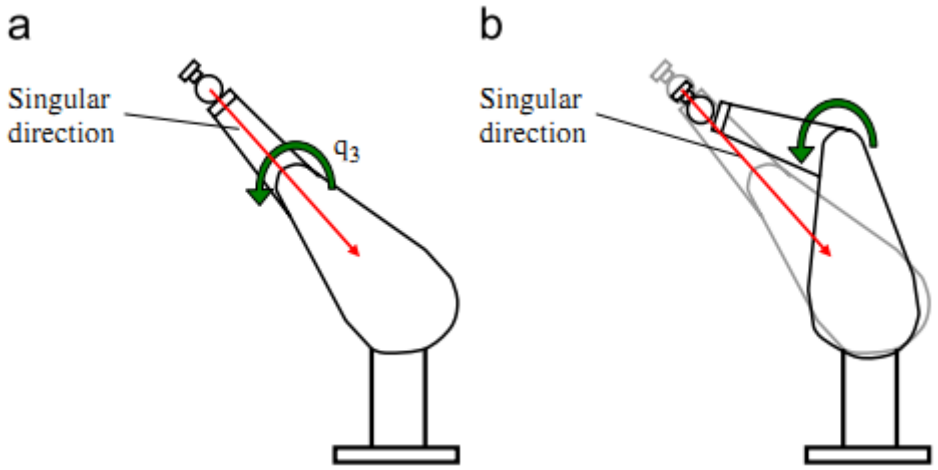

Figure 13 - Puma560 "elbow lock"

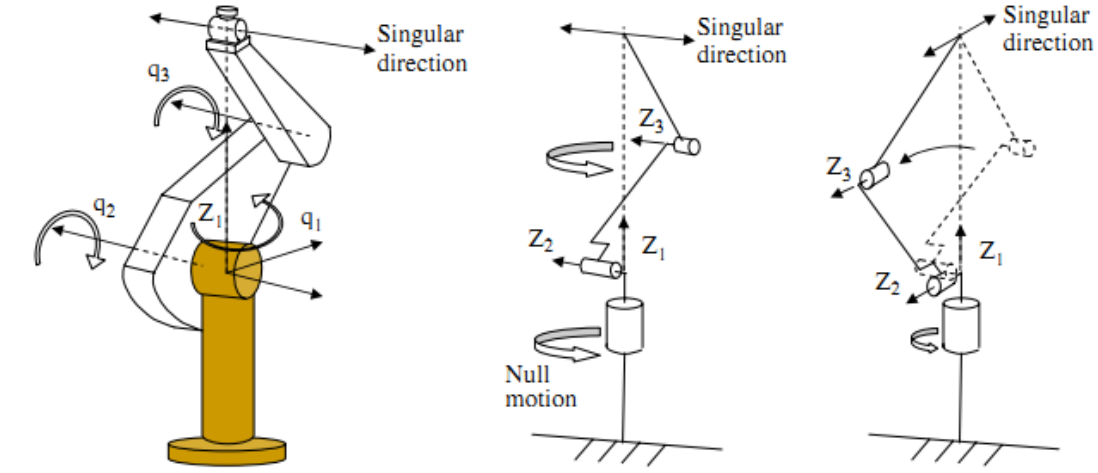

Figure 14 - Puma560 "head lock"

3.2. NOTESnail singularities study

The Puma560 robot has a very important particular configuration that allows it to decouple "arm" and "wrist". That procedure becomes highly useful given that the distance between joints at the wrist are much smaller than those distances at the arm.

Therefore the Puma560 architecture implies that positioning will be a task majorly achieved by the first three joints and orientation will be majorly done by the three last joints, which are called "spherical joint". A spherical joint has also the special propriety that all three rotation axis intersect in a common point.

Since NOTESnail has the same number of joints, all from the same type (rotational), a similar approach of decouple position and

orientation could be hypothesized. Between the 4th and the 5th joints, the distance is negligible, but the distance between the 5th and the 6th joints correspond to one third of the robot's total length. That means that position and orientation will be tasks performed by all 6 joints.

From this point forward, it is convenient to distinguish orientation and aiming. Using Denavit-Hartemberg convention, it is recommended that the z-axis at the end-effector coordinate frame equals to the "attack" direction, which in our case also equals to the last module direction, as shown in Figure 15. Therefore, for the propos of this thesis, **aiming** will be defined by the end-effector z-axis, expressed in the base coordinate frame. **Orientation** will be given by the end-effector rotation matrix, expressed in the base coordinate frame.

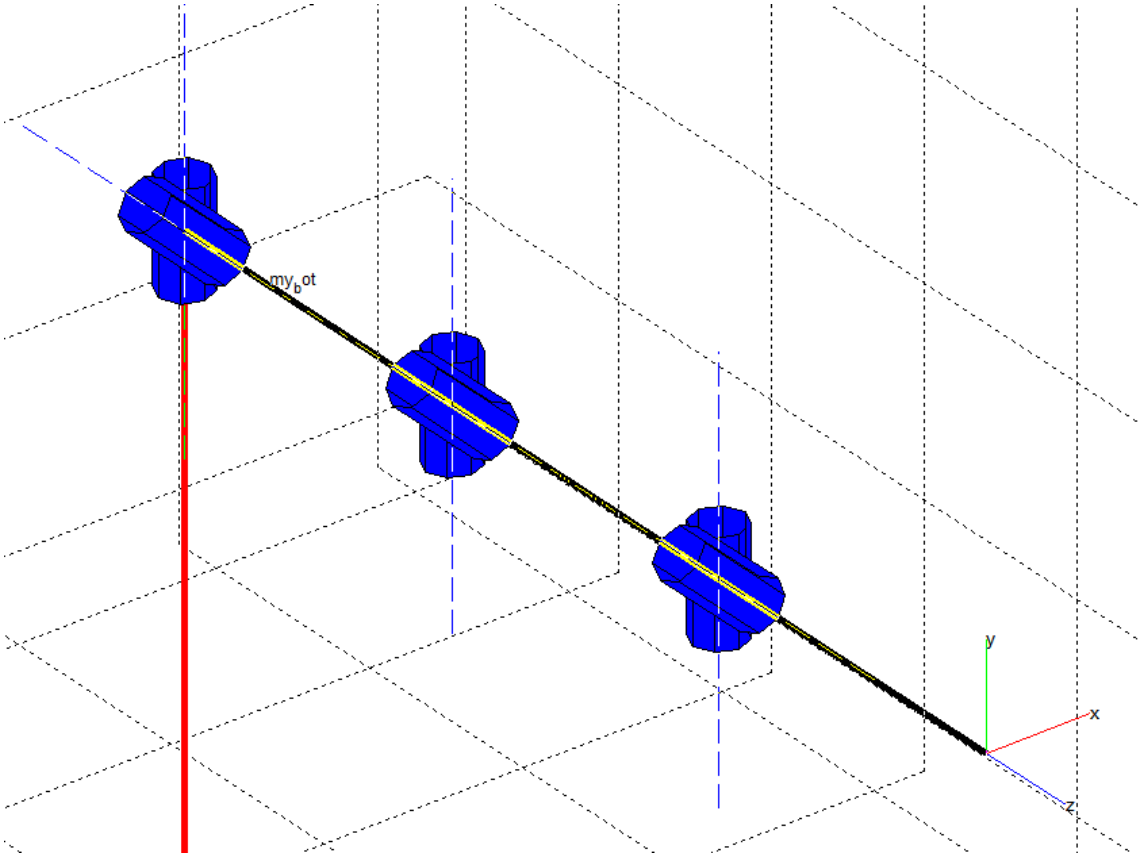

Figure 15 - MATLAB NOTESnail representation

Analyzing the homogenous transforms between coordinates systems between the 6th joint and the end-effector (shown in Equation 3), it

may be inferred that the 6th joint plays no rule at positioning or aiming of the end-effector. That last revolution joint is the surgeon easiest resource for correctly orienting a grasper, for example.

$$A_6^e = \begin{matrix} \cos\theta_6 & -\sin\theta_6 & 0 & 0 \\ \sin\theta_6 & \cos\theta_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

Equation 3

It is assumed that the 6th joint range of motion will be capable of providing all necessary orientations for the surgeon in a given aiming; therefore, this last joint will be ignored in the analysis of (sections x-y).

3.2.1. Modeling workspaces with analytical geometry

Each of NOTESnail modules has two joints: a bending mechanism is the proximal part and a torsion mechanism in the distal part. In this section, two different workspaces will be modeled: as if the end-effector was placed at the end of the 1st module and as if it was at the end of the 2nd module.

These models will be represented by an analytic geometry equation and a graphical representation. Both models will be useful to check if a certain configuration is reachable, evaluating singularities, building optimization algorithms and more, as shown in the next sections.

The first workspace, corresponding to the 1st and 2nd joints is shown in Figure 16, and it's equation may be approximated by:

$$x^2 + y^2 = d^2$$

Equation 4

Where *d* is the module's length and Equation 4 exists only if $y \leq 0$.

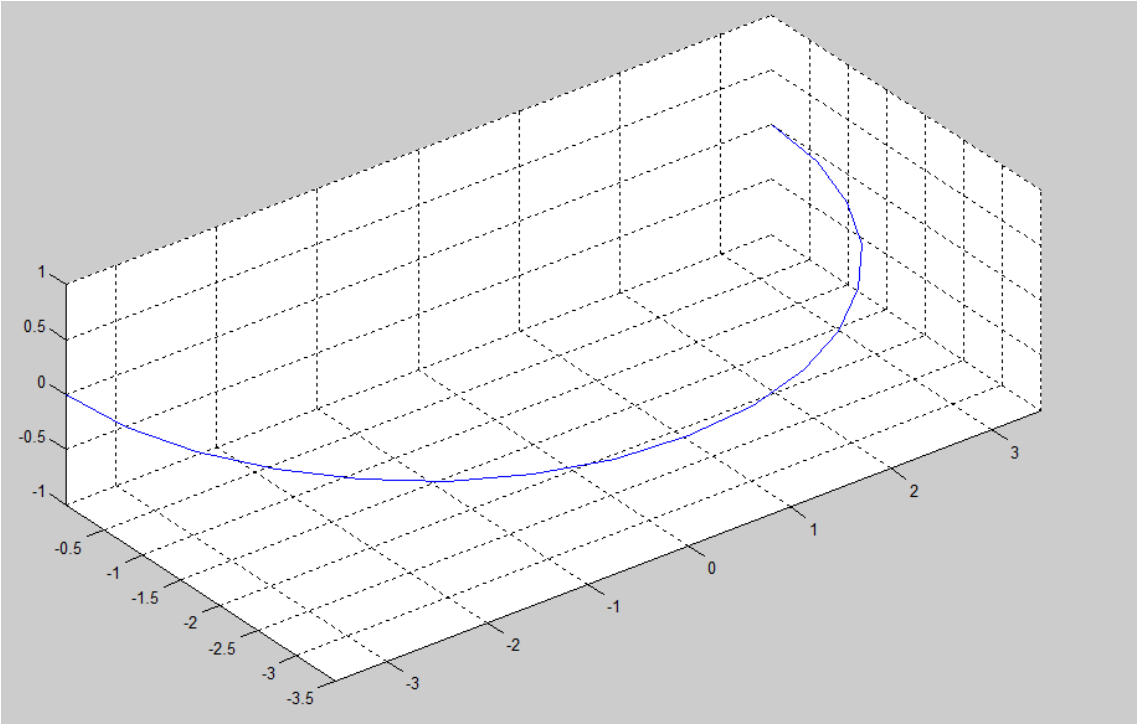

Figure 16 - First module workspace

The second workspace is a solid. Initially, the combination of the 2nd and 3rd joints actions on the second module would build a surface in the shape of half-sphere, centered at one point at the first module workspace (given by the 1st joint angle) as shown in Figure 17. If the 1st joint angle is varied within its range, the end result is shown in Figure 18 and Figure 19.

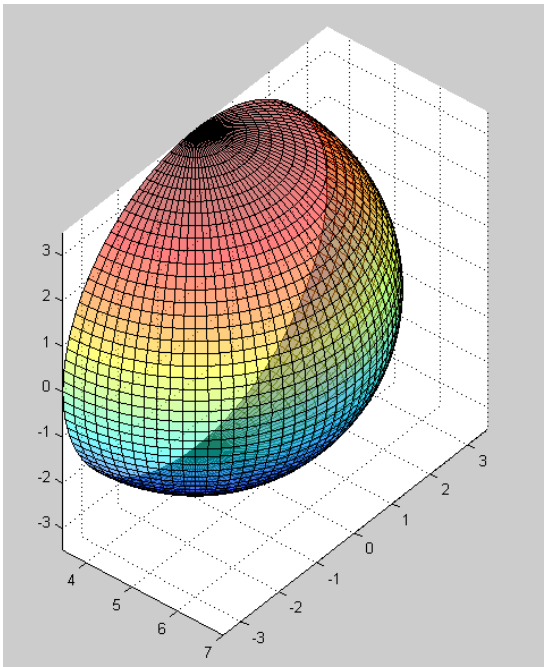

Figure 17 - 2nd and 3rd joints action with the 1st joint set to zero

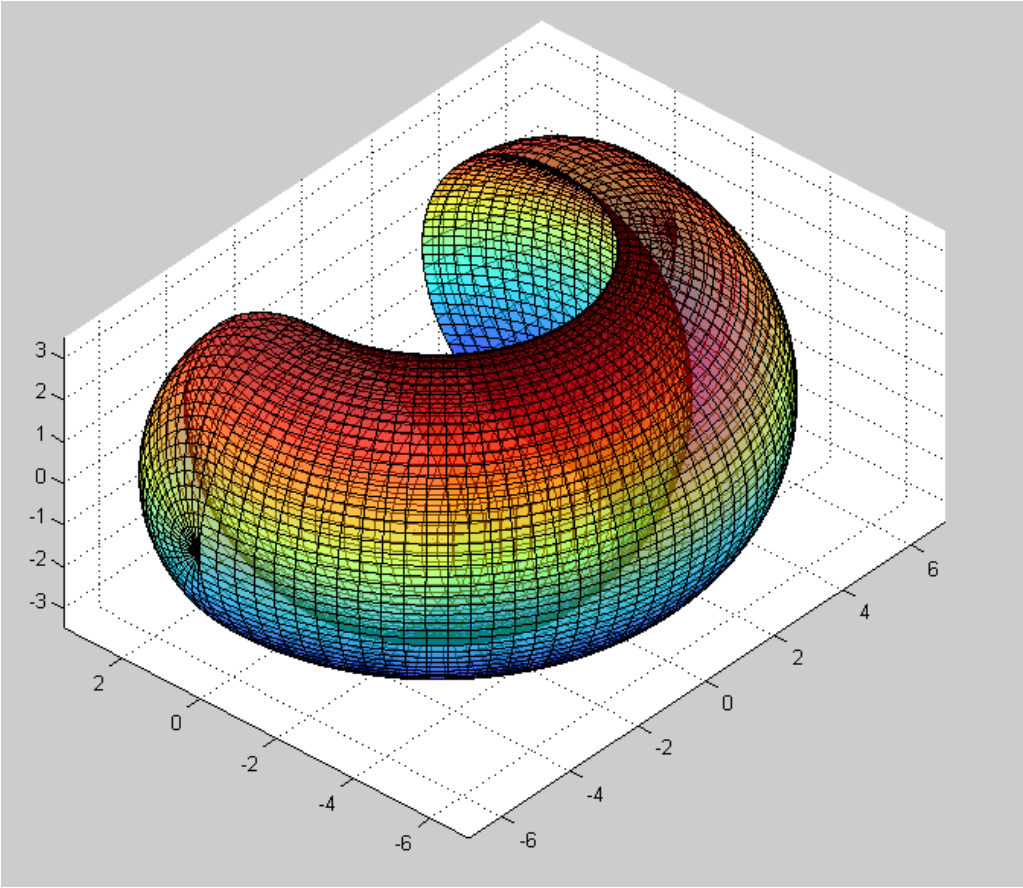

Figure 18 - 2-module workspace

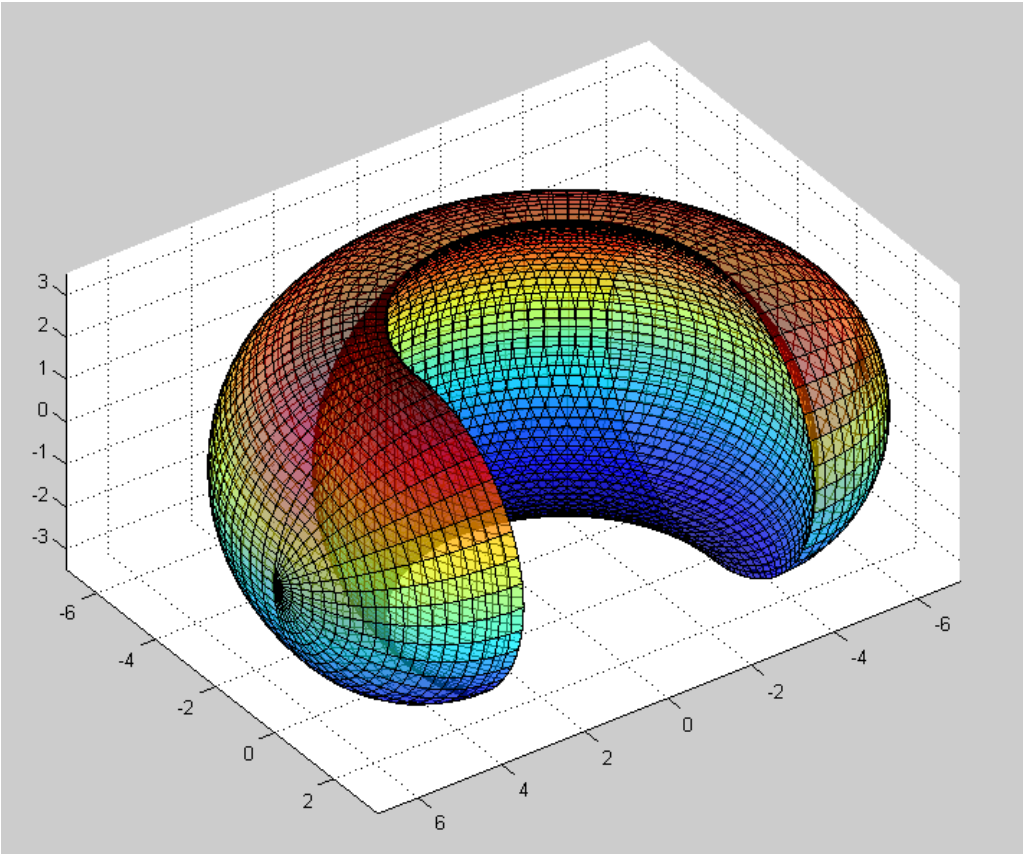

Figure 19 - 2-module workspace

The solid can be approximated by a torus (approximation only valid for $y \leq d$ and $x^2 + y^2 \geq d^2$), but its general equation is given by Equation 5.

$$x^2 + y^2 + z^2 = 2d\sqrt{x^2 + y^2}$$

Equation 5

3.2.2. Reachable coordinates and measure of redundancy

As pointed in earlier sections, this analysis will be based on positioning and aiming, ignoring the effects of the 6th joint.

If the end-effector is anchored at a certain point in space, all the possible aiming vectors summed would build a sphere solid centered at that certain point (represented in Figure 20 by the yellow point). Taking account that the aiming vector has the same direction as the module's longitudinal axis, that sphere radius is equal to the module's length. As a result, the relevant geometric model is a surface in the shape of a sphere centered in the end-effector desired point with radius equal to the module's length (represented in Figure 20 by the red sphere). That geometric model is equivalent to the Equation 6.

$$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = d^2$$

Equation 6

Where (x_0, y_0, z_0) are the desired Cartesian coordinates of the end-effector and *d* is the module's length.

Conclusion 1: If the sphere surface intersects the torus-like solid, then (x_0, y_0, z_0) is reachable.

Conclusion 2: The intersection between the sphere surface and the torus-like solid is a spherical cap.

Conclusion 3: The spherical cap area corresponds to the number of the NOTESnail possible configurations (meaning all possible aiming) with the desired positioning (as shown in Figure 20). That figure will be called **measure of redundancy.**

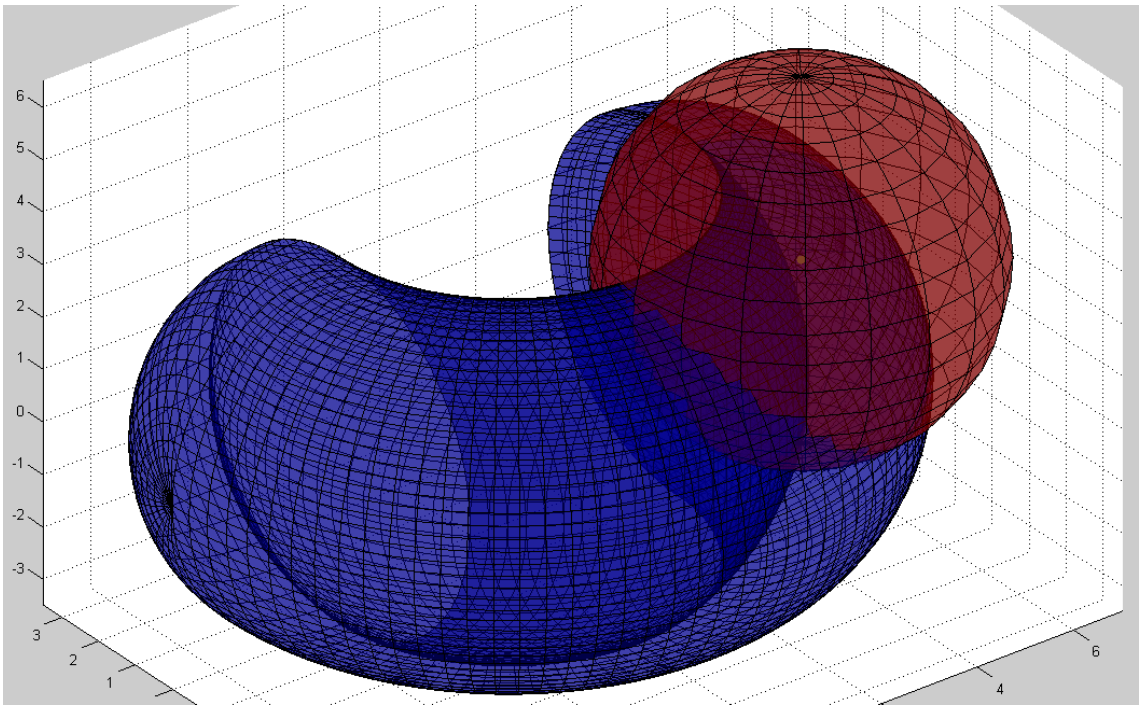

Figure 20 – Reachable point (yellow dot)

3.2.3. NOTESnail singular configurations

The singular configurations, as introduced in the section 3.1, were found using screw theory

3.2.3.1. 2nd joint – type I singularity

The 2nd joint being equal to +π/2 or –π/2 represent a type I singularity because it represents the frontier of the 2-module workspace. The last module is an important part on determining the end-effector's orientation and the job of positioning the end-effector becomes a task of the first two modules.

To analyze the possible motions when the singular configuration has been reached, the best coordinate system to be used is the one shown in Figure 21. Because of the frontier interpretation, it is clear that any movement that is tangent to the 2-module workspace would be possible (represented by $\hat{\theta}$ and $\hat{\Phi}$). In the $\hat{r}$ negative direction, any motion is still allowed, but the positive direction of $\hat{r}$ is impossible, as well as any other trajectory that contains that vector as a partial component.

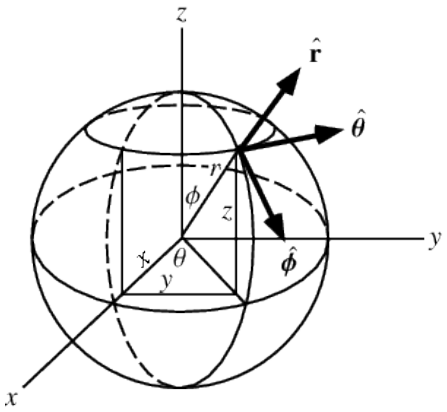

Figure 21 - Spherical coordinates

3.2.3.2. 3rd joint – type I singularity

The 3rd joint singularity also represents a boundary singularity, as shown in Figure 22, but physically, it also represents the annulation of the effects of the 2nd joint, given the mechanical architecture shown in Figure 15. It is important to notice that the 3rd joint singularity is contained in the 2nd joint singularity, both in physical and mathematical meaning, which means that a singularity avoidance algorithm for the 2nd joint will be also effective for the 3rd joint.

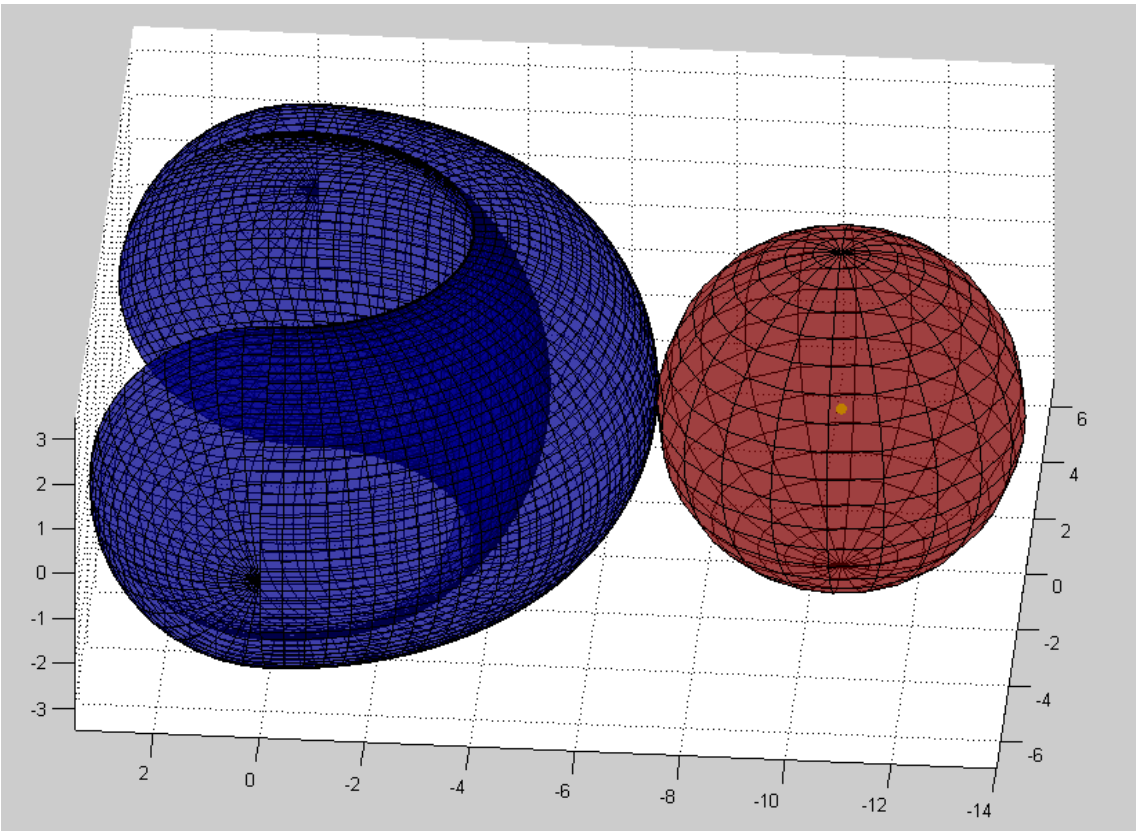

Figure 22 - 3rd joint singularity

3.2.3.3. 5th joint – type II singularity

The 5th joint singularity, given by ($\theta_5 = 0$) represents the physical alignment of the 2nd and 3rd module and is the only one between the ones studied so far that is perfectly avoidable by using an alternative configuration, as shown in Figure 23. This implies that when approximating a singular configuration, the desired control algorithm would create an internal motion to reach the alternative configuration without passing through the singularity, as shown in Figure 23, Figure 24, Figure 25 and Figure 26.

From this point, some problems have emerged:

- Given the robot's joint limits, not all singular configurations have a symmetrical non-singular configuration, as the one shown in Figure 27
- Depending on the rotation angle that the robot is introduced in the patient's body, the rotation joints limits may represent an issue to internal motion.
- Having a symmetrical non-singular configuration would require the control system to be coordinated by a navigation system, which would be complex and independent from the user. That navigation system would have to be capable of measuring the end-effector's position and orientation in order to input the commands to create the internal motion capable of avoiding the singularity. That system is not difficult to design, but its necessity should be well evaluated according to the behavior of the inverse kinematics algorithms.

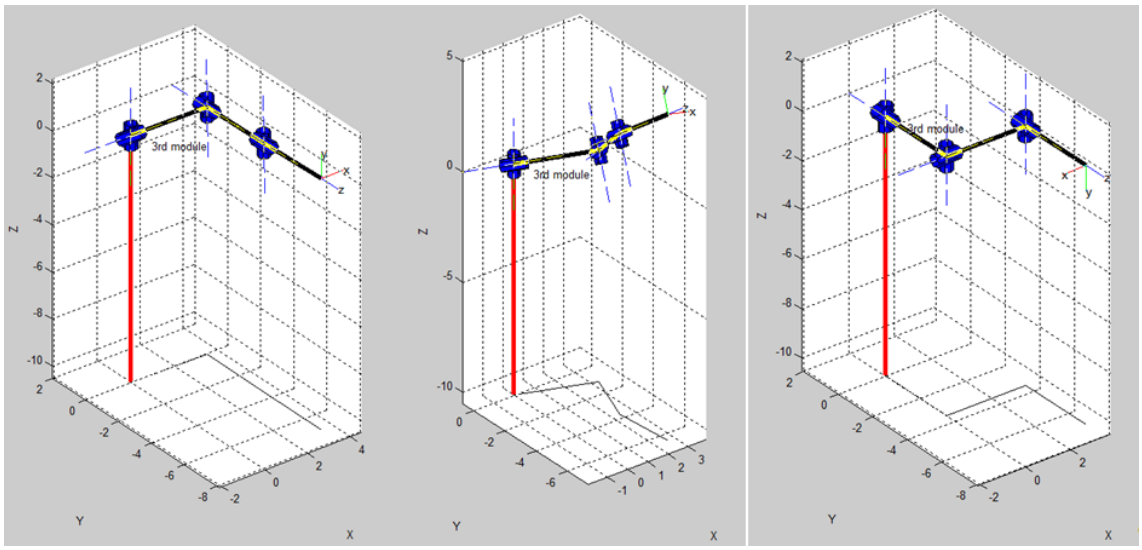

Figure 23 - Motion from singular to alternative configuration

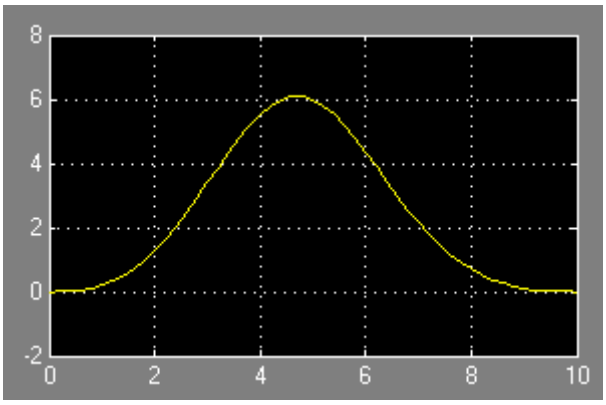

Figure 24 – Time vs. z coordinate during singularity avoidance movement

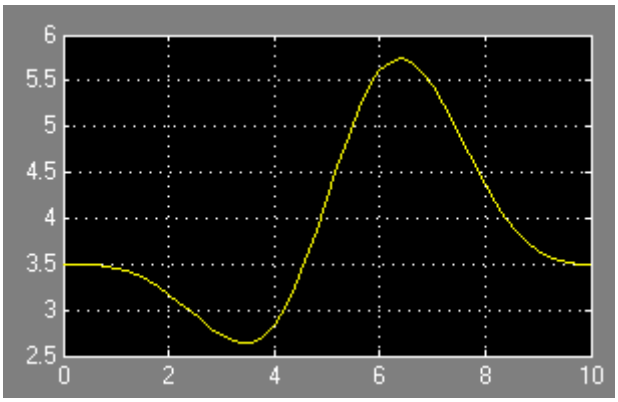

Figure 25 - Time vs. x-coordinate during singularity avoidance movement

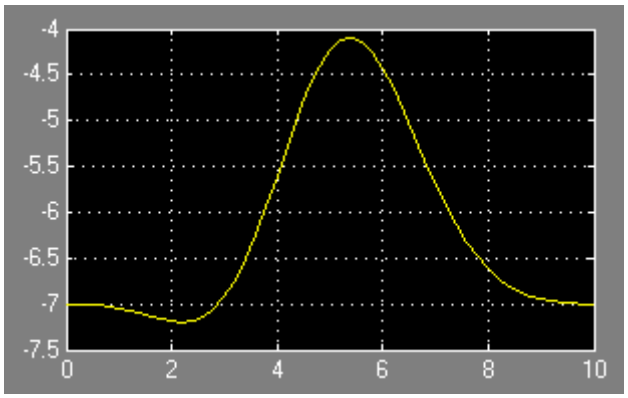

Figure 26 - Time vs. y-coordinate during singularity avoidance movement

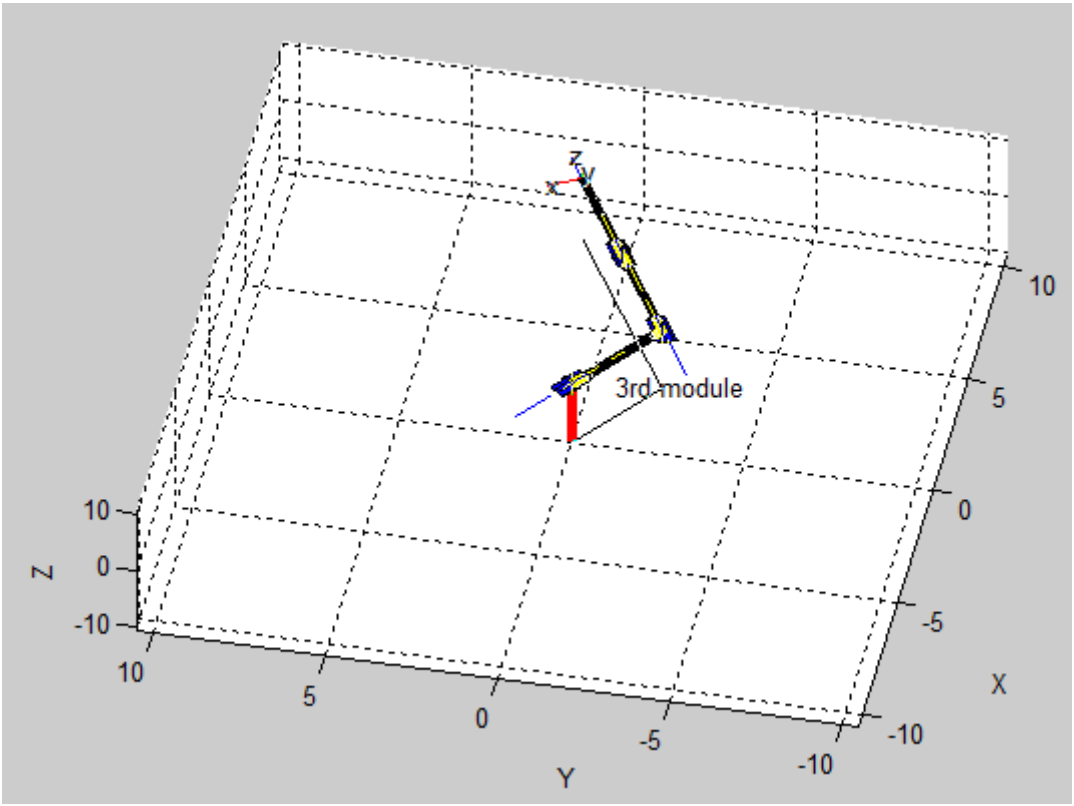

Figure 27 - Singular configuration without non-singular reachable symmetrical

4. Robotics Control Algorithms

Starting from the general control theory, a controller manipulates the dynamic system inputs in order to obtain a desired output called "reference". An open-loop control system does not measure outputs in order to alter control. On the other hand, a closed-loop control system has sensors whose measures are used to drive dynamic changes in the overall system. (Ogata, 2001)

A closed-form control system is usually represented in a block diagram, as shown in Figure 28. The most common objectives of a control algorithm are:

- Disturbance robustness;
- Improve performance around uncertain models;
- System stabilization;

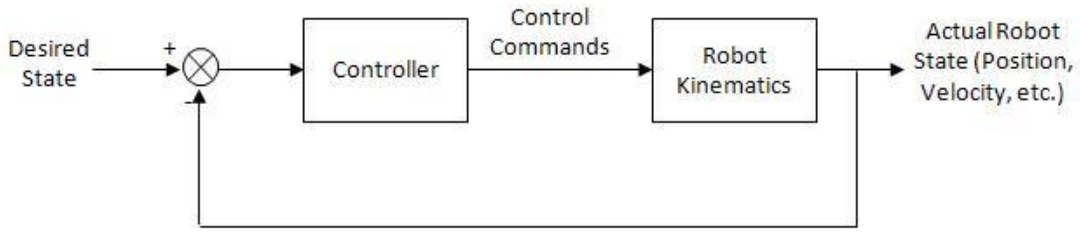

Figure 28 - Control system block diagram

There are many control techniques and methodologies that can be applied to the control of manipulators. The particular control method chosen as well as the manner in which it is implemented can have a significant impact on the performance of the manipulator and consequently on the range of its possible applications. For example, continuous path tracking requires a different control architecture than does point-to-point control. (Spong, Hutchinson, & Hutchinson, 2004)

In addition, the mechanical design of the manipulator itself will influence the type of control scheme needed. For example, the control problems encountered with a Cartesian manipulator are fundamentally different from those encountered with an elbow type

manipulator. In sections 0 to 4.3, some control schemes (or their important components) are presented.

Since every control scheme is based on a reference value (given as an input), an important paradigm that needs to be explored is the trajectory planning versus real-time human-controller input. The first is majorly used in many industrial and autonomous robots, since their tasks are programmable and will be executed in repeatedly.

In trajectory planning (or generation), the basic problem is to move the manipulator from an initial configuration to a final configuration (maybe via some other configuration). The trajectory will be a set of positions, velocities and accelerations for each degree of freedom. The constrains may be spatial, temporal and/or about smoothness.

The solutions for the trajectory generation problem may be in the joint space or in the Cartesian space.

While giving a solution in the joint space has the advantages of singularity avoidance and less calculations, the solution may not follow a straight line. On the other hand, a solution in the cartesian space has many discontinuity problems such as unreachable and singular configurations along the path.

Many authors have proposed solutions for a smooth, fast and reliable path generation using mathematical interpolation or genetic algorithms (Abo-Hammour, Mirza, Mirza, & Arif, 2002). Some of these solutions include singularity and/or obstacle avoidance, very useful for pre-programmed tasks.

The problem is that the surgeon controlling NOTESnail will give real-time inputs to the system in the Cartesian space, which is the intuitive space for humans. The reference and the "sensor feedback" will be both be given by the surgeon, therefore, the electronic control will be necessary only for stabilizing the system

4.1. Computed torque control

This is a classical dynamic control technique when the rigid-body dynamic model is inverted to compute the demand torque for the robot based on current joint angles and joint angle rates and demand joint acceleration (Paul, 1981).

In the case of the simulation toolkit used in this work, the inverse dynamics of NOTESnail is calculated using the recursive Newton-Euler algorithm. As the dynamics parameters are not exactly known, the toolkit's author recommends the introduction of a perturbance. (Corke, 2008)

However, computer torque control algorithms have as an input a trajectory (or a desired position) given in the joint space, which is undesirable for our user. For moving from the Cartesian space (user-friendly) for the joint space, both inverse kinematics and inverse Jacobian functions are required.

4.2. Visual servoing control

Visual servoing is the concept in which machine vision can provide closed-loop position control for a robot end-effector. The task is to use visual information to control the pose of the robot's end-effector relative to a target object or a set of target features. (Hutchinson, Hager, & Corke, 1996)

Visual servoing depends on a series of other subjects, such as: coordinate transformations; velocity of a rigid object; camera projection models; image features and image features parameter space; and camera configuration (end-effector mounted or fixed in the workspace)

There is a taxonomy, introduced by (Sanderson & Weiss, 1980), which categorizes visual servo systems into 4 different schemes (shown in Figure 29 to Figure 32) by answering two questions:

- Is the control structure hierarchical, with the visual system providing set-points as input to the robot's joint-level controller,

or does the visual controller directly compute the joint-level inputs?

- Is the error signal defined in 3D (task space) coordinates, or directly in terms of image features?

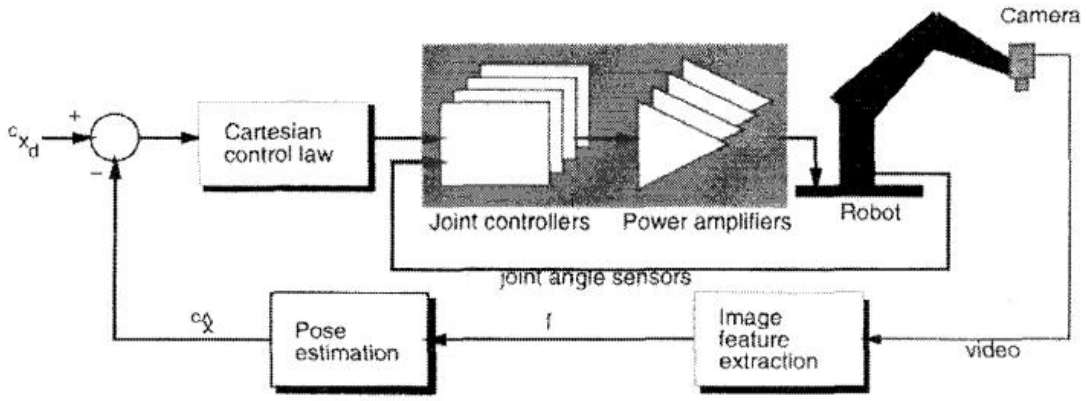

Figure 29 - Dynamic position-based look-and-move structure

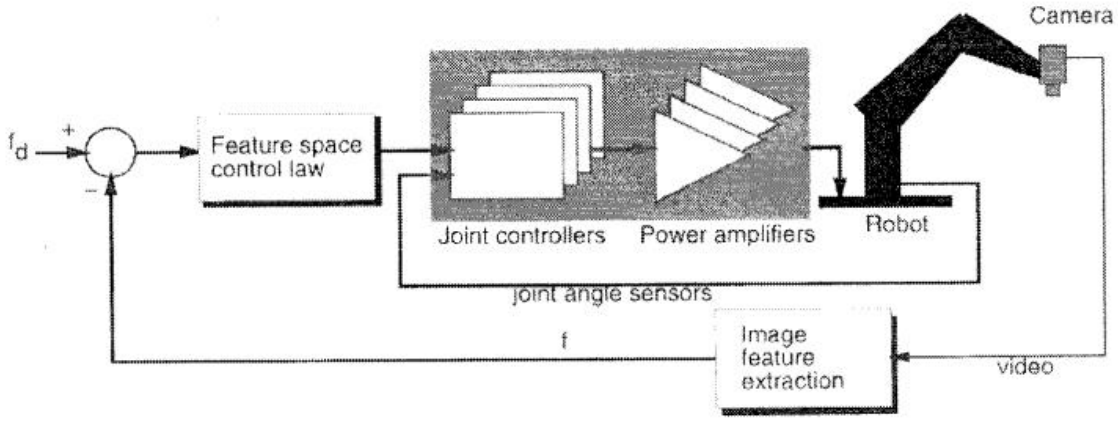

Figure 30 - Dynamic image-based look-and-move structure

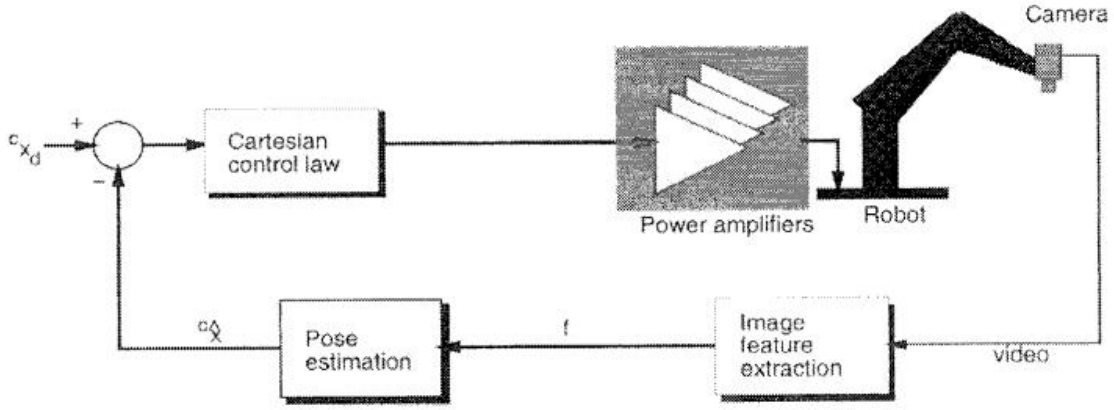

Figure 31 - Position-based visual servo

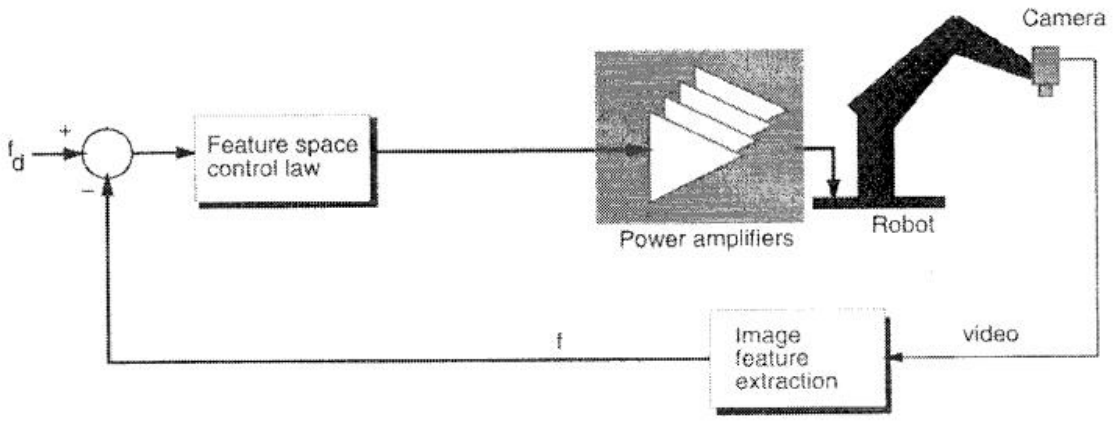

Figure 32 - Image-based visual servo

Today, the applications of this technology remain limited due to the high costs of technology and the skills required to construct an integrates visually controlled robot system (Hutchinson, Hager, & Corke, 1996) , but still is a very useful tool for NOTESnail.

4.3. Cartesian control

An important objective in controlling a robot manipulator is to ensure that the end-effector tracks a desired Cartesian (task) space trajectory as closely as possible. This is a difficult problem because, in the description of the manipulator dynamics, the control torques are defined in terms of joint variables and not end-effector variables. (Misra, Patel, & Balafoutis, 1988)

Manipulator control is therefore usually performed in joint space. The desired task space trajectory is transformed to joint space at sufficiently many points to ensure that any error in tracking the joint space trajectory will not cause the end-effector to deviate significantly from the desired trajectory in task space. However, since the relationship between joint variables and end-effector variables is, in general, highly nonlinear, it is possible that a small deviation in a joint space trajectory can cause a relatively large deviation in the corresponding task space trajectory.

The basic scheme for a Cartesian control uses:

- The **reference** is a trajectory given in end-effector homogenous transforms;
- The **error** is the difference between the reference and the actual position (given by the joint angles encoders plus a forward kinematics algorithm);
- This error is transformed in a (6x1) **differential vector** equivalent to a column vector of Cartesian velocities;
- This vector is multiplied by the inverse of the manipulator Jacobian. The result is a column vector of **joint velocities**;
- If this vector of joint velocities is integrated, the actual joint configuration is achieved and the feedback the system, as presented in Figure 33.

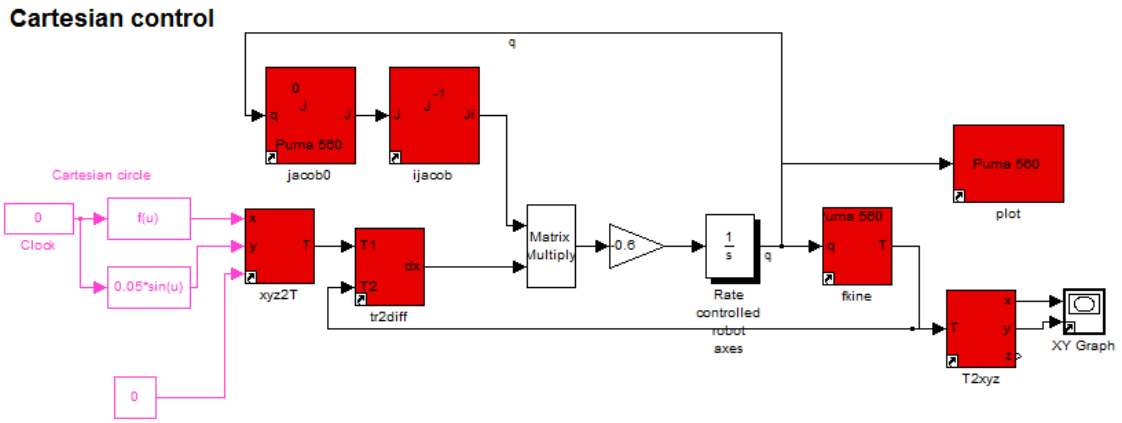

Figure 33 - Basic Cartesian Control Scheme

In the basic scheme presented, it is noticeable the importance of the singular configurations during the operation of the Jacobian inverse. Since the simple matrix inversion does not exist near singularities, other methods should be explored in order to make the Cartesian control well-behaved near singular configurations.

These algorithms, which basically solve the inverse kinematics problems, are vastly explored in the literature and include: cyclic coordinate descent methods (Wang & Chen, 1991); pseudoinverse methods (Whitney, 1969); Jacobian transpose methods (Balestrino, de Maria, & Sciavicco, 1984); damped least squares methods (Wampler, 1988); quasi-Newton and conjugate gradient methods

(Wang & Chen, 1991); and neural nets and artificial intelligence methods (Oyama, Chong, Agah, & Maeda, 2001).

For the propose of this work, four inverse kinematics algorithms were selected using the following criteria: simplicity of implementation, computational cost, real-time application and adequacy to the task (since many of the algorithms are designed for computer graphic applications, which have a larger number of degrees of freedom, more complicated constrains, etc.) (Tolani, Goswami, & Badler, 2000).

For simplicity, it won't be considered any aspects of self-collision or joint limits (except for the optimization algorithm).

4.3.1. Jacobian transpose method

The basic idea is very simple: use the transpose of J instead of the inverse of J . That is, we set Δθ equal to:

$$\Delta\theta = \alpha J^T e;$$

Equation 7

for some appropriate scalar α. Now, of course, the transpose of the Jacobian is not the same as the inverse; however, it is possible to justify the use of the transpose in terms of virtual forces. (Balestrino, de Maria, & Sciavicco, 1984)

4.3.2. Jacobian pseudo-inverse method

The pseudoinverse method sets the value Δθ equal to:

$$\Delta\theta = J^* e;$$

Equation 8

where the *n* x *m* matrix J^* is the *pseudoinverse* of J , also called the Moore-Penrose inverse of J . This inversion can be used for all matrices, even those which are not full rank. The pseudoinverse gives the best possible solution to the equation $J\Delta\theta = e$ in the sense of least squares.

(Buss, 2004) affirms that:

Let Δθ be defined by Equation 8. First, suppose *e* is in the range (i.e., the column span) of J . In this case, $J\Delta\theta = e$; furthermore, Δθ is the unique vector of smallest magnitude satisfying $J\Delta\theta = e$. Second, suppose that *e* is not in the range of J. In this case, $J\Delta\theta = e$ is impossible. However, Δθ has the property that it minimizes the magnitude of the difference $J\Delta\theta - e$. Furthermore, Δθ is the unique vector of smallest magnitude which minimizes ||JΔθ-e||, or equivalently, which minimizes $||J\Delta\theta-e||^2$.

The pseudoinverse is not immune to stability problems in the neighborhoods of singularities. If the configuration is exactly at a singularity, then the pseudoinverse method will behave well and won't try a movement in an impossible direction. However, if the configuration is close to a singularity, then the pseudoinverse method will lead to very large changes in joint angles, even for small movements in the target position.

The pseudoinverse has the further property that the matrix $(I - J^*J)$ performs a projection onto the nullspace of J. Therefore, for all vectors φ, $J(I - J^*J)\varphi = 0$. This means that we can set Δθ by

$$\Delta\theta = J^*e + (I - J^*J)\,\varphi$$

Equation 9

for any vector φ and still obtain a value for Δθ which minimizes the value J Δθ - e. By choosing special values of φ, it is possible to achieve secondary goals other that following to a demanded trajectory. For example, φ can be set to return the joint angles back to rest positions (Girard & Maciejewski, 1985): this can help avoid singular configurations.

A number of authors have used the nullspace method to help avoid singular configurations by maximizing Yoshikawa's manipulability measure (Yoshikawa, 1985). (Maciejewski & Klein, 1985) used the nullspace method for obstacle avoidance.

4.3.3. Damped least squares method

The damped least squares method can be theoretically justifed as follows: Rather than just finding the minimum vector Δθ that gives a best solution to equation e = J Δθ, we find the value of Δθ that minimizes the quantity

$$||J\,\Delta\theta - e||^2 + \lambda^2||\,\Delta\theta||^2,$$

Equation 10

where λ ∈ R is a non-zero damping constant. By using singular value decomposition, it can be shown that $J^TJ+\lambda^2I$ is non-singular. Thus, the damped least squares solution is equal to:

$$\Delta\theta = (J^TJ + \lambda^2 I)^{-1}J^T\vec{e}$$

Equation 11

Now, JTJ is a n x n matrix, where n is the number of degrees of freedom. It is easy to show that $(J^TJ+\lambda^2I)^{-1}J^T = J^T(JJ^T+\lambda^2I)^{-1}$. Then,

$$\Delta\theta = J^T(JJ^T + \lambda^2 I)^{-1}\vec{e}$$

Equation 12

The advantage of Equation 12 over Equation 11 is that the matrix inversion is executed over a m x m matrix instead of a n x n (m being the number of degrees of freedom and n being the number of joints, which is usually larger).

4.3.4. Singular Value Decomposition method

Let A be a real *m* x *n* matrix with $m \geq n$:

$$A = U\Sigma V^T$$

Equation 13

Where:

$$U^TU=V^TV=VV^T=I_n \text{ and } \Sigma=diag(\sigma_1,\ldots, \sigma_n).$$

The matrix U consists of n orthonormalized eigenvectors associated with the *n* largest eigenvalues of AA^T, and the matrix V consists of the orthonormalized eigenvalues of A^TA. The diagonal elements of Σ are the non-negative square roots of the eigenvalues of A^TA; they are called *singular values*. It is assumed that:

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$$

Thus, if *rank*(A)=r, $\sigma_{r+1}= \sigma_{r+2}= \ldots = \sigma_n=0$. The decomposition presented in Equation 13 is called the *singular value decomposition* (SVD). (Golub & Reinsch, 1970)

One of the applications of the SVD procedures is the calculation of the pseudoinverse X, that satisfies the following four properties:

- A X A = A;
- X A X = X;
- $(A\ X)^T = A\ X$;
- $(X\ A)^T = X\ A$.

The unique solution is denoted by A^+. It is easy to verify that if $A=U\Sigma V^T$, then $A^+=V\Sigma^+U^T$ where $\Sigma^+=diag(\sigma_i^+)$ and

$$\sigma_i^+ = \frac{1}{\sigma_i} for\ \sigma_i > 0$$

$$\sigma_i^+ = 0\ for\ \sigma_i = 0$$

4.3.5. Optimization method

There are many optimization algorithms available for the inverse kinematics calculation. For the propos of this work, it was chosen the minimization of constrained non-linear multivariable function. The variables to be optimized are the joint angles and the function to me minimized is:

$$f(q) = abs(norm(tr - fkine(robot, q)))$$

Where "abs" calculates the absolute value inside the parenthesis, "norm" calculates the norm of the matrix inside the parenthesis and "fkine" calculates the forward kinematics of the "robot" while in the configuration set by "q".

In comparison with inverse kinematics methods that use any variation of the Jacobian matrix, the optimization method has the vantage of respecting the joint limits, but the disadvantage of higher computational cost. Both methods have problems near singularities, depending on the choice of the initial guess.

5. Experimental Results

When dealing with surgical applications of robots, specially having a human input for the manipulator's motion, it is very important to have a singularity-robust motion control, in order to avoid tracking errors, blockages or any other undesired behavior that would risk a patient surgery.

For better choosing a motion control algorithm, a series of tests were conducted in order to analyze the behavior of each method presented in sections 4.3.1 to 4.3.5. The testing protocol involves a series of steps from building a trajectory to collecting the results, which will be presented in the next few sections.

5.1. Trajectories

The trajectories were chosen in two different perspectives: in the first, one should try to mimic possible motions a surgeon would execute inside the operating room; in the second, the robot's behavior in extreme singularity conditions have to be tested, even if those situations are not applicable in a real world application.

As mentioned in previous sections, simulated trajectories have to be set in the Cartesian space, since those are the natural command inputs for the human controller. But, to make sure that those trajectories are composed only of reachable configurations, they are all tested through a optimization algorithm.

This testing algorithm uses the same objective function of the inverse kinematics in section 4.3.5, but returns a time-stamped array of joint space coordinates and a time-stamped array of the objective function value at that given set of coordinates. In other terms, gives a measure of the error during the trajectory.

All the following testing trajectories have a tracing error inferior to 10^{-5} meters and the norm of the maximum linear velocity is inferior to 0.5 cm/s.

5.1.1. Pick and Pull

The pick-and-pull trajectory is a common trajectory to be used in the operating room. Literary hundreds of times the surgeon has to grab a piece of tissue or suture line and pull it in other direction. The proposed trajectory follows a straight line along the positive direction of z-axis.

The orientation is kept parallel to the X-Z plane with a 45 degrees rotation around the negative axis of Y, as shown in Figure 34. The total distance traveled is 2 centimeters, starting from [3;-5;-4,5]. The trajectory in joint coordinates is shown in Figure 35:

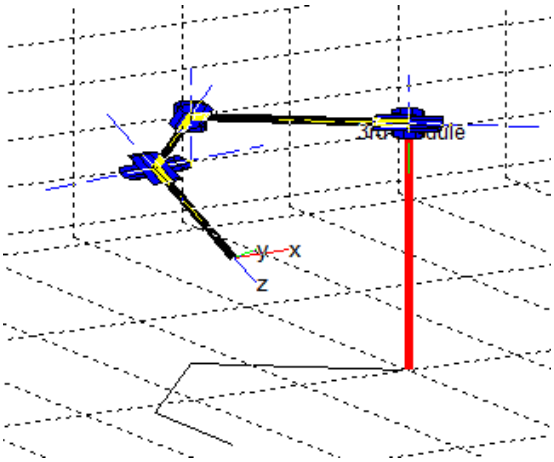

Figure 34 - Trajectory 1 final configuration

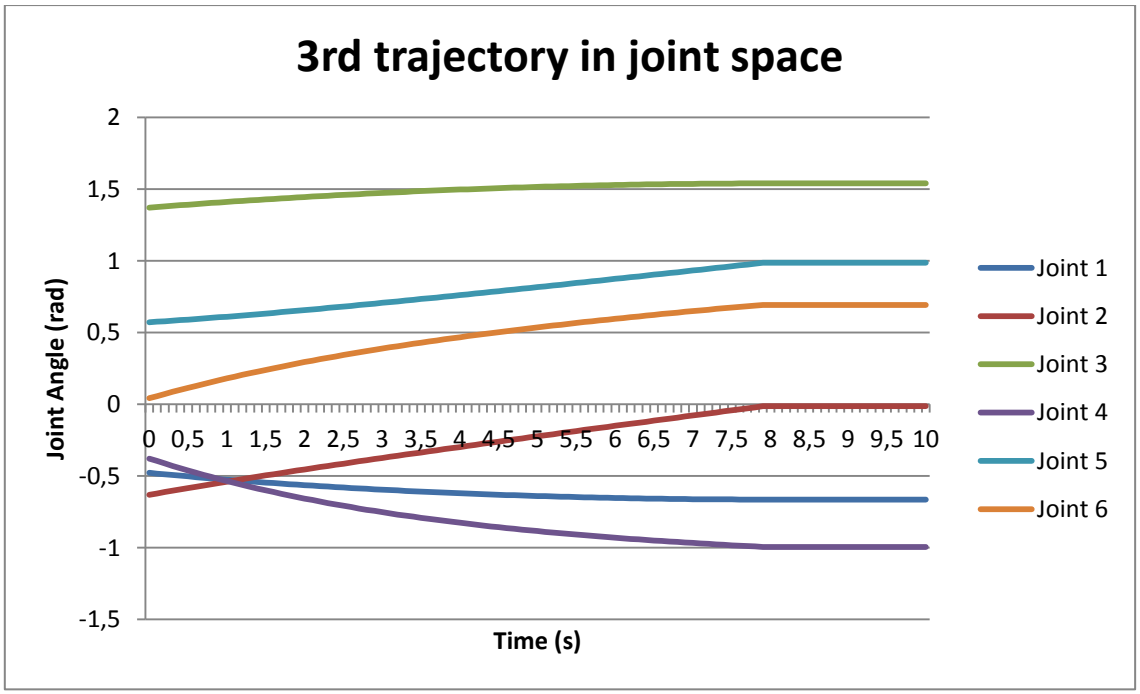

Figure 35 - Trajectory 1 joint space trajectory

5.1.2. Type I singularity test trajectory

This is a trajectory that is not comparable to real-world applications, but is important as an extreme condition test. In this situation, the starting point was not a Cartesian coordinate route: the procedure was to block angles for joints 1, 3, 4, 5 and 6 while varying the 2nd joint within its full range.

That causes NOTESnail to pass through a type I singularity twice: when q2 equals to –п/2 and to –п/2, as shown in (same legend as previous equivalent figures). The Cartesian coordinates visualization is provided in.Figure 36, Figure 37 and Figure 38.

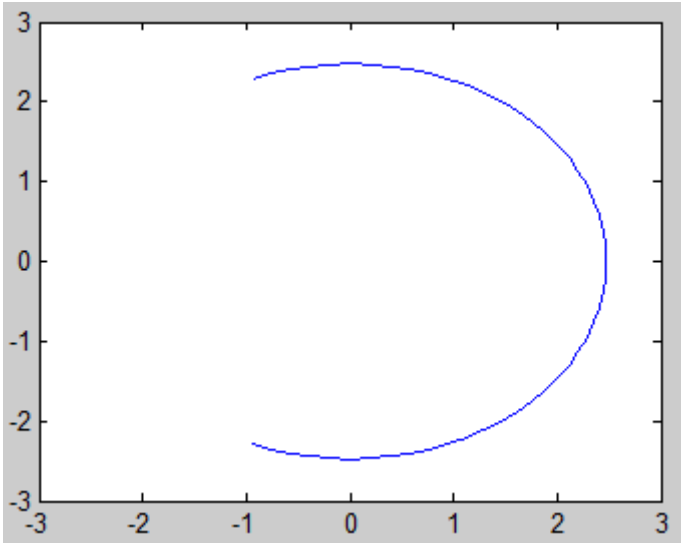

Figure 36 - X-Z graphic of trajectory 5.1.2

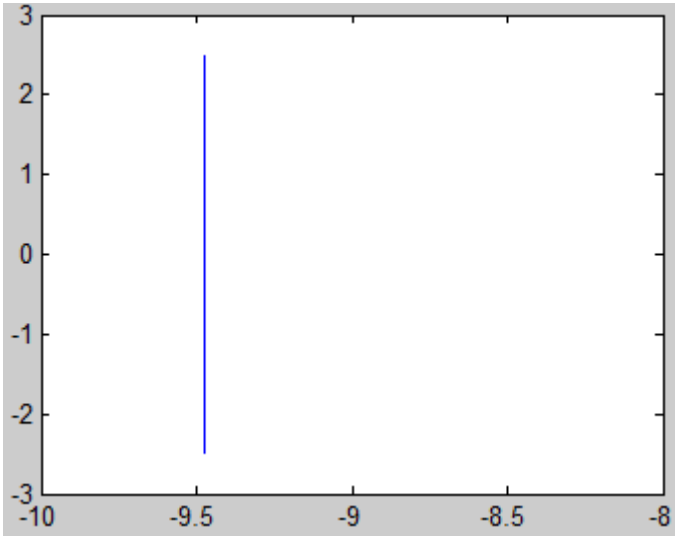

Figure 37 - Y-X graphic of trajectory 5.1.2

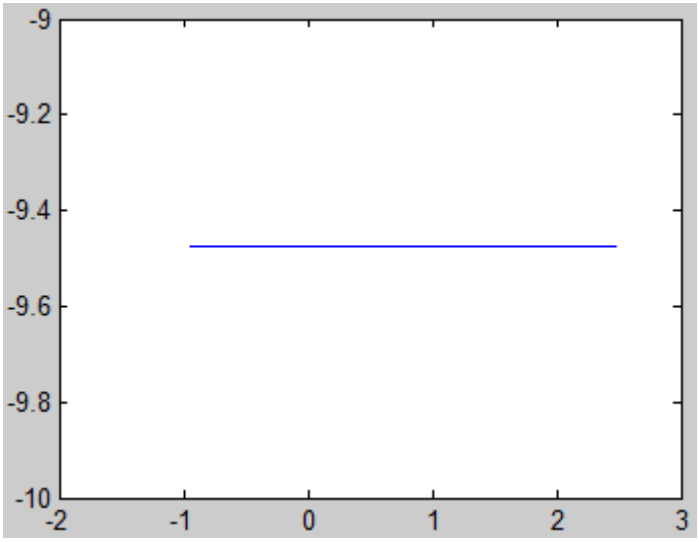

Figure 38 - - Z-Y graphic of trajectory 5.1.2

5.1.3. Type II singularity test trajectory

This trajectory follows the same principle of trajectory 5.1.2 concerning real-world application, but instead of varying the 2nd joint variable, the 5th joint singularity is the one to be studied.by using the exactly equivalent procedure. The joint space variables visualization is shown in Figure 39 (note that some joint coordinates can't be visualized since all have the same value "0").

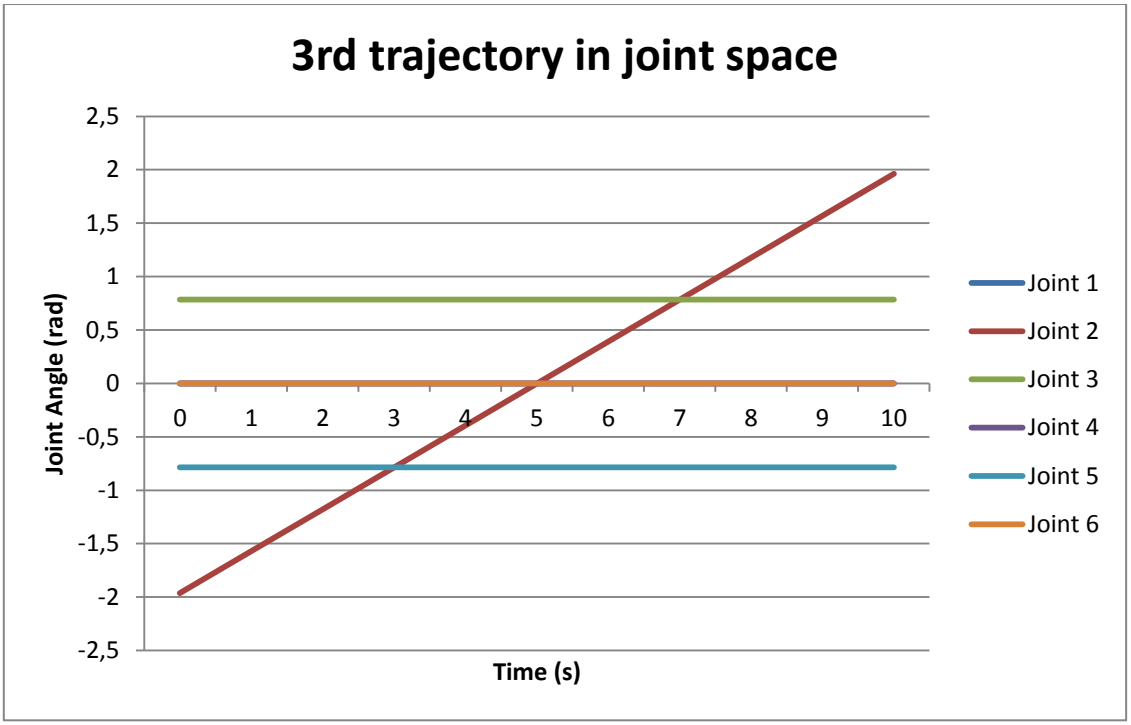

Figure 39 - Joint space trajectory of 3rd trajectory

5.1.4. Suture simulation

Another trajectory vastly used in the operating theater is the suture, as shown in Figure 40. The movements shown in the sections A, C, D, E and F are achieved with the previous "pick and pull" movement. The most complicated movement is shown in Figure 40-B when the surgeon does and helix-like trajectory in order to "wrap" the suture line around the grasper body.

That movement can be mathematically approximated with an helix around the end-effector z-axis. According to the common surgery practice, instead of keeping the orientation constant during the complete trajectory, it will be used to help the circular movement. Joints 1, 2 and 3 will be used for the linear movement along the last module z-axis. Joints 4 and 5 will be used for the circular movement of the helix.

The trajectories in the joint space and in the Cartesian space are shown in Figure 41 and Figure 42 respectively. In Figure 42, the colors yellow, magenta and cyan represent the Cartesian coordinates x, y and z respectively.

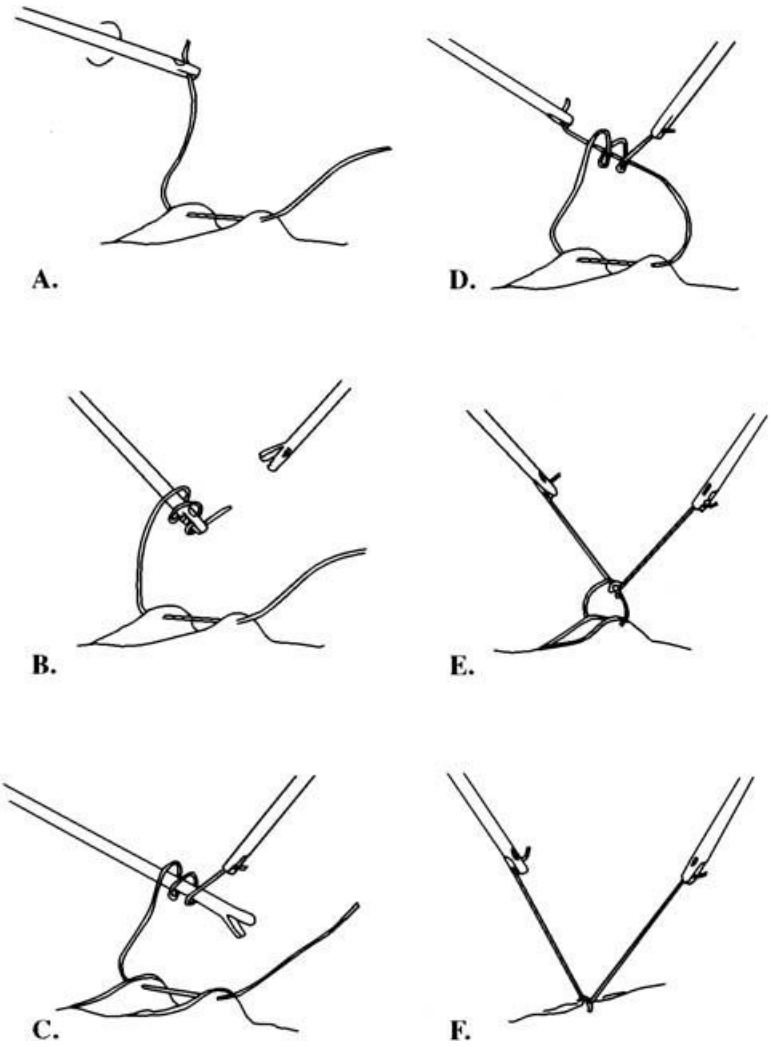

Figure 40 - Laparoscopic suture

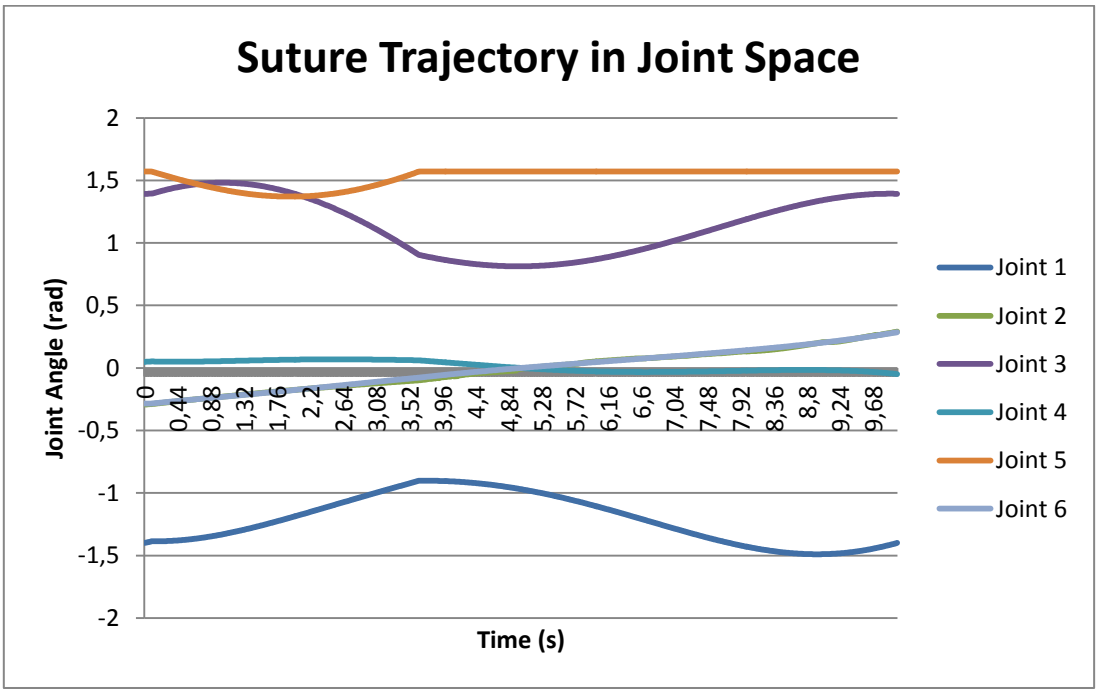

Figure 41 - Second trajectory in joint space coordinates

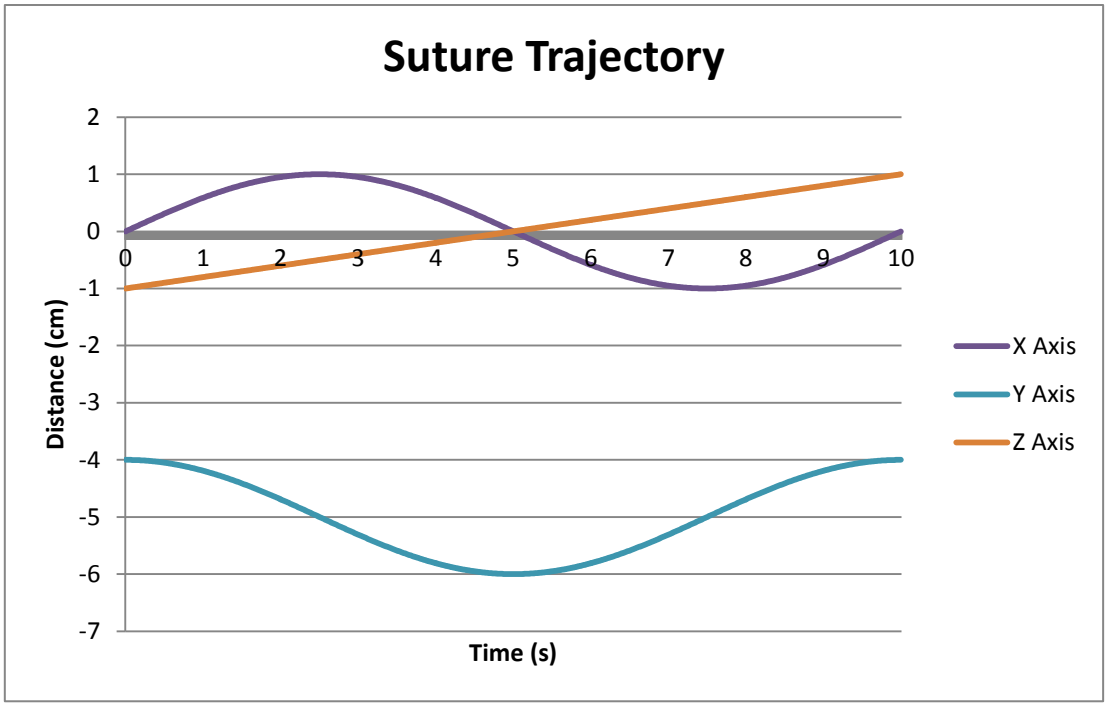

Figure 42 - Suture trajectory in Cartesian coordinates

5.2. Simulink models

Simulink, developed by Mathworks is a tool for modeling, simulating and analyzing dynamic systems. Its primary interface is based in block diagrams, making it easy to change architectures, parameters and etc. That was the main reason for choosing it for the propos of this work.

In all models, the blocks are the following:

- the Cartesian trajectory is obtained from a MATLAB file (resulted from a "trajectory building" file);
- the difference between the actual position and the desired position is computed in the form of a (6x1) differential vector;
- this Cartesian differential vector is transformed in a joint space differential vector according to one of the methods;
- A proportional integrative operation is performed. The gain can be altered as desired;
- The forward kinematics operates the variables back to the Cartesian space;

- The "plot" block provides the user to a visualization of the movement.

5.2.1. Transpose Method

The transpose method is self-explanatory. The joint variables are taken in each step for the calculation of the manipulator's Jacobian, which will be transposed in order to multiply the Cartesian differential vector.

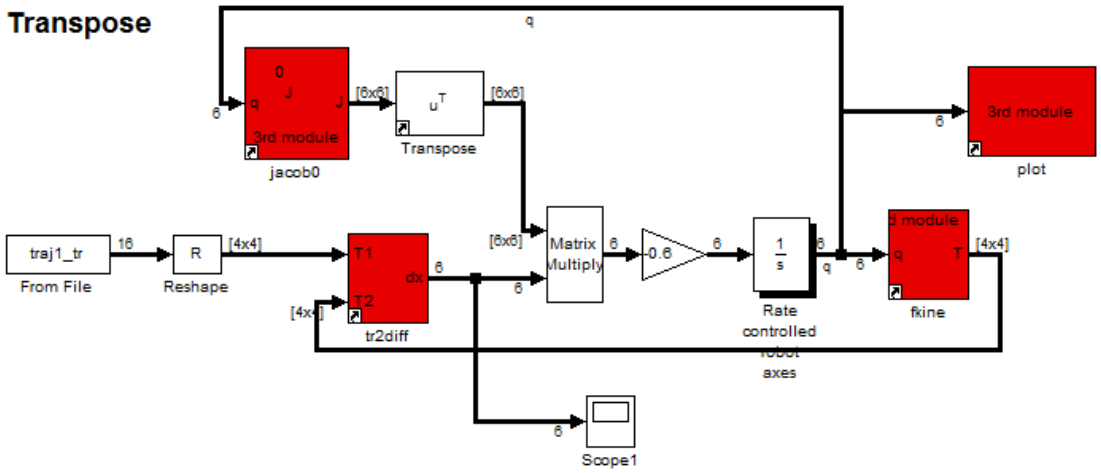

Figure 43- Simulink model of the transpose method

5.2.2. Pseudoinverse method

The pseudoinverse method introduces a larger number of block in order to perform the operations described in the section 4.3.2. The block named "Subsystem1" in Figure 44 has a user-determined input called "null-space vector" in order to perform such an operation.

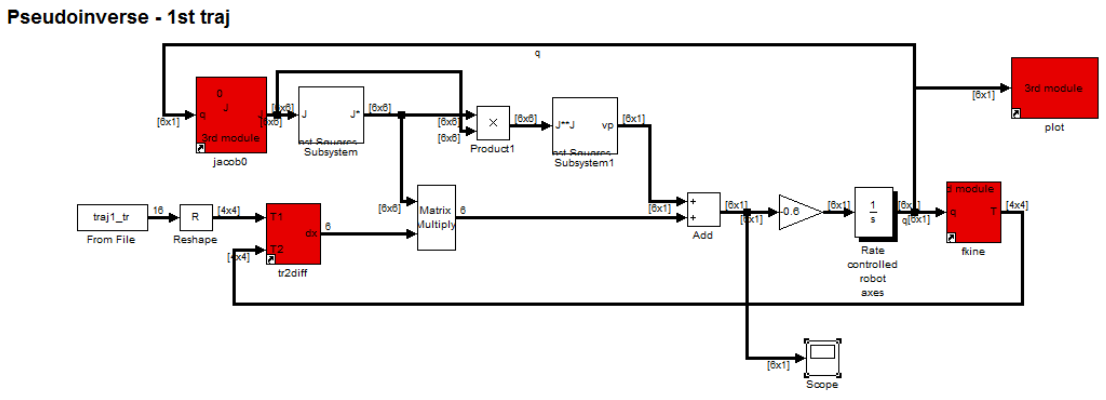

Figure 44 - Pseudoinverse method simulink method

5.2.3. Damped least squares method

This Simulink model is also self-explanatory. The mathematical operations proposed in the section 4.3.3 is performed with a user-defined input for the damping constant.

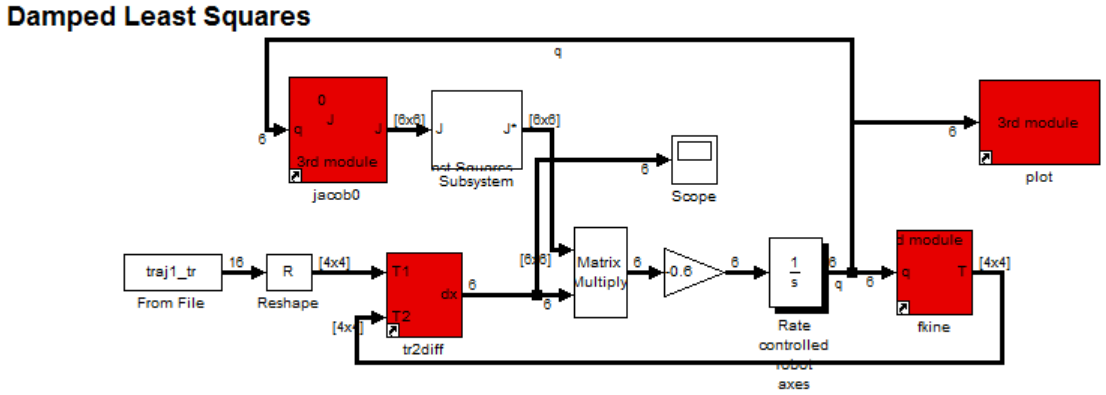

Figure 45 - Damped least squares method simulink model

5.2.4. Singular value decomposition method

For the SVD method, a MATLAB function was created instead of combining many Simulink® blocks, making the overall model clearer. This function can be found in Appendix A

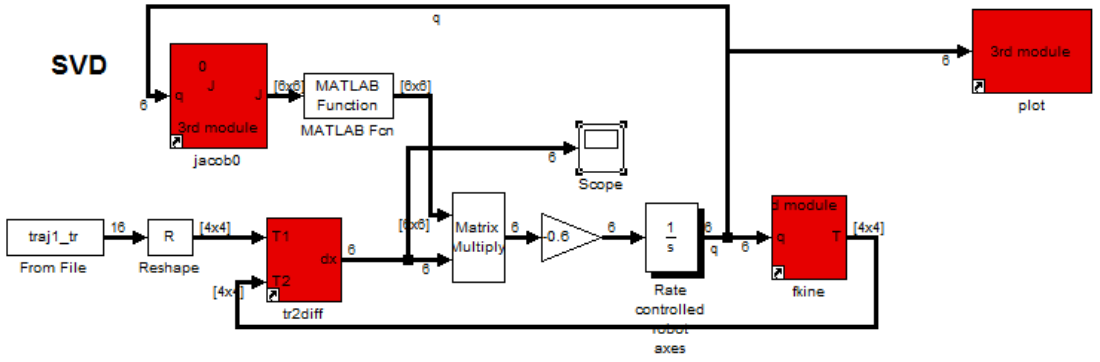

Figure 46 - SVD method simulink method

5.2.5. Optimization method

The optimization method has the least self-explanatory Simulink model (Figure 47). The block transforming the difference between the actual state and the desired trajectory in a differential vector was

kept because that is the user's play in the system. The differential vector being the joystick input.

The optimization code represented by "ikinebot_full" is shown below:

```
function [qt yout] = ikinebot_simulink (robot,diff,qi)

liminf = [-pi/2 -3*pi/4 -pi/2 -3*pi/4 -pi/2 -3*pi/4];
limsup = [pi/2 3*pi/4 pi/2 3*pi/4 pi/2 3*pi/4];
% compute final configuration
ti = fkine(robot,qi);
di = tr2diff(ti);
df = di+diff;
tr = diff2tr(df);
%optimization algoritm
f = @(x)full_matrix_op(x,robot,tr);
[qt yout] = fmincon(f,qi,[],[],[],[],liminf,limsup);

end
```

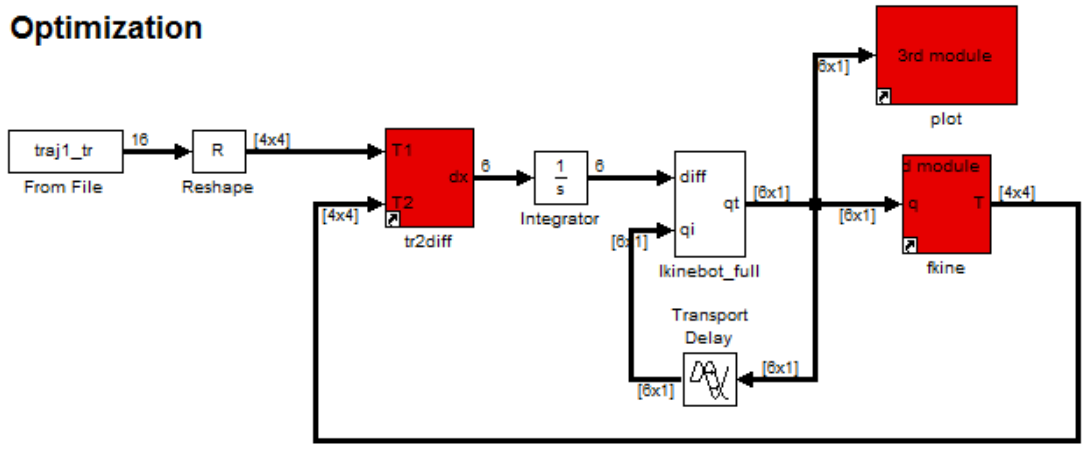

Figure 47 - Optimization simulink model

5.3. Graphical Cartesian space offset

For all figures in this section, the horizontal axis corresponds to the time (in seconds) and the vertical axis corresponds to the following legend is valid for all figures in section 5.3:

- Dark Blue = linear error in x-axis
- Green = linear error in y-axis
- Red = linear error in z-axis
- Cyan = rotational error around x-axis
- Magenta = rotational error around y-axis

- Yellow = rotational error around z-axis

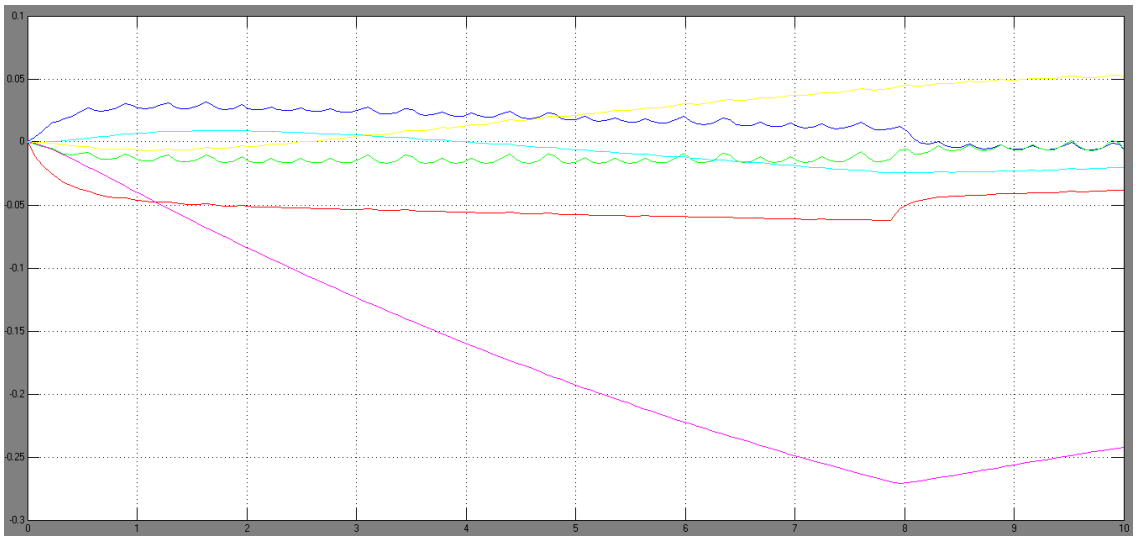

Figure 48 - Trajectory 1, transpose method

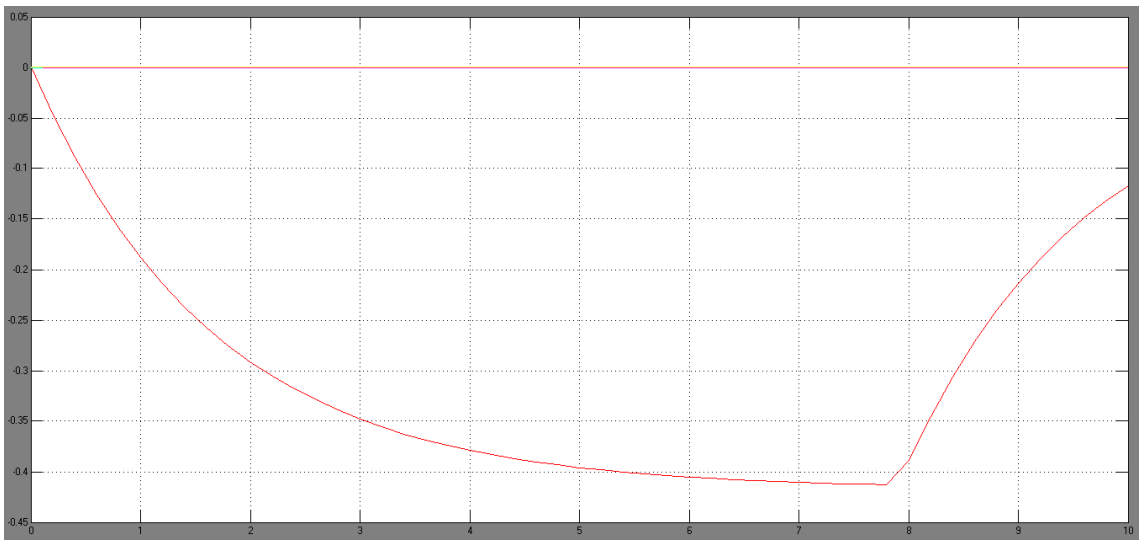

Figure 49 - Trajectory 1, DLS

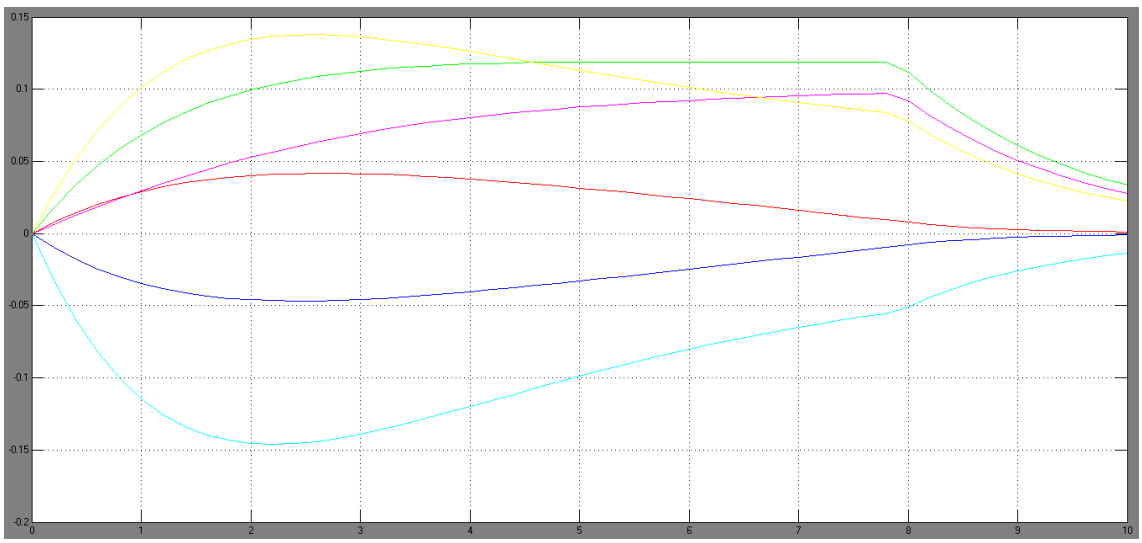

Figure 50 - Trajectory 1, pseudoinverse

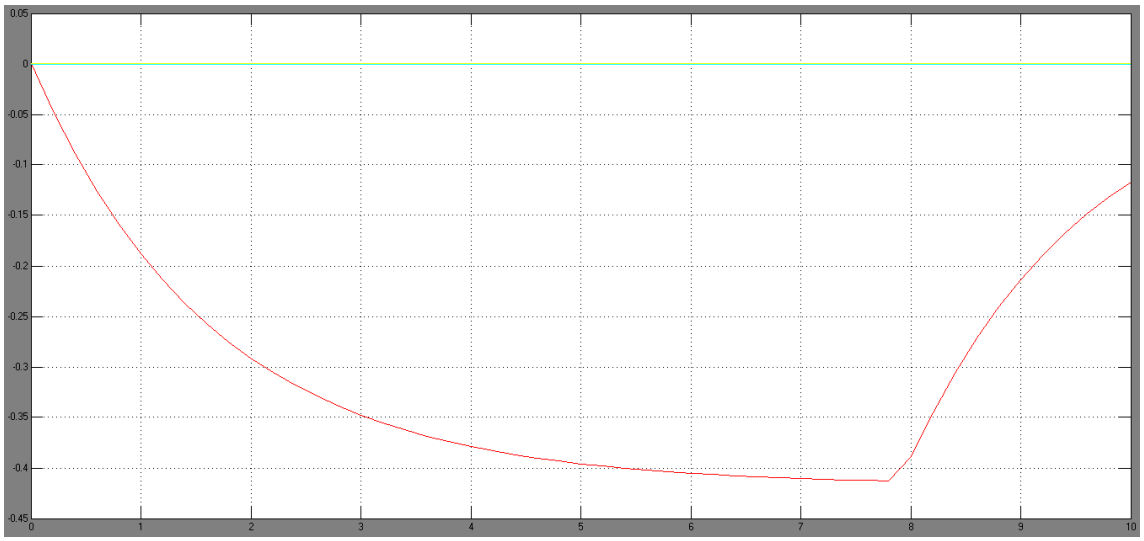

Figure 51 - Trajectory 1, SVD

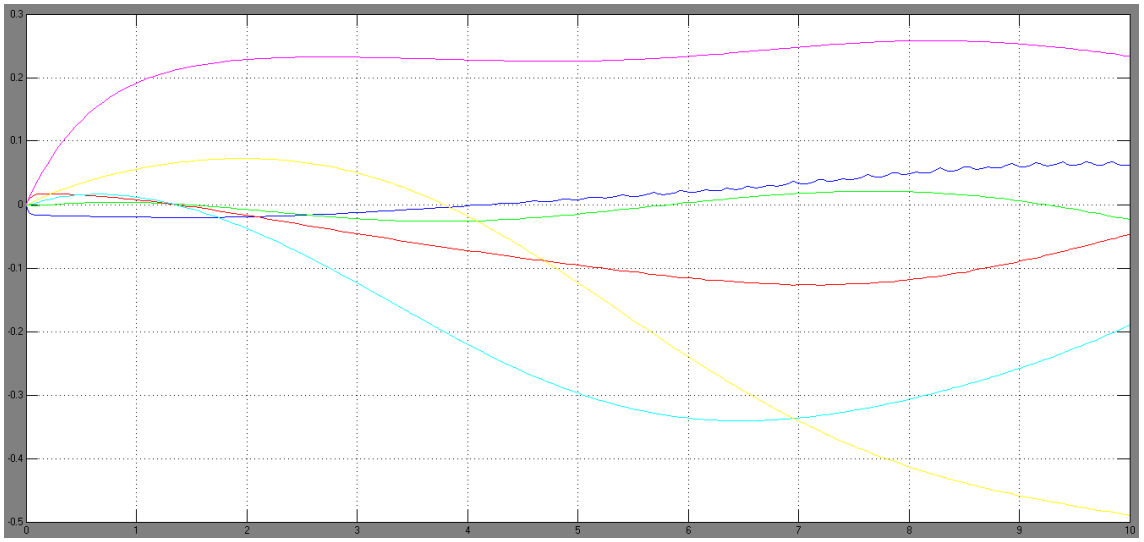

Figure 52 - Trajectory 2, transpose method

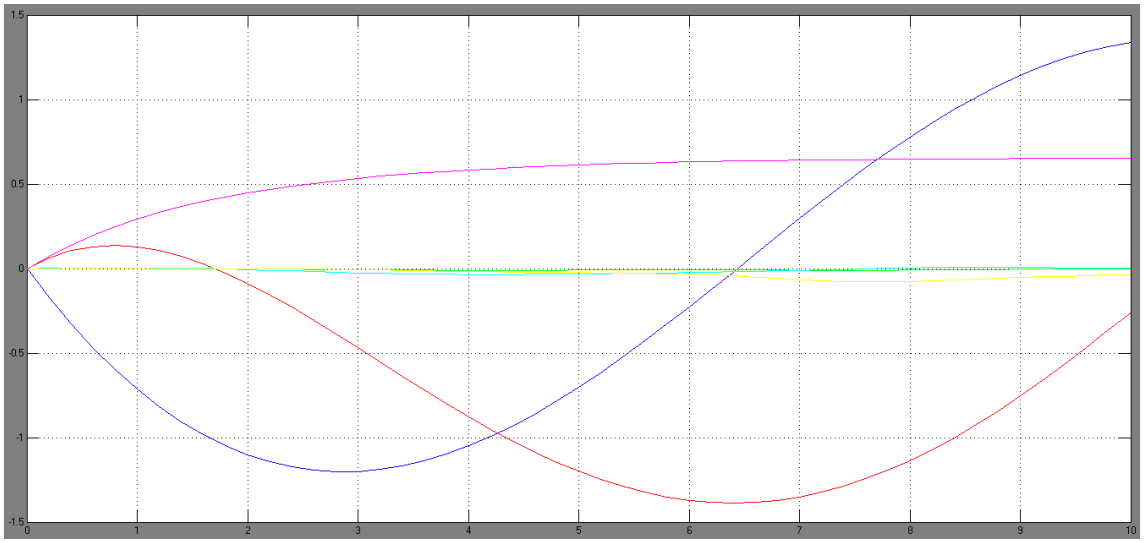

Figure 53 - Trajectory 2, DLS method

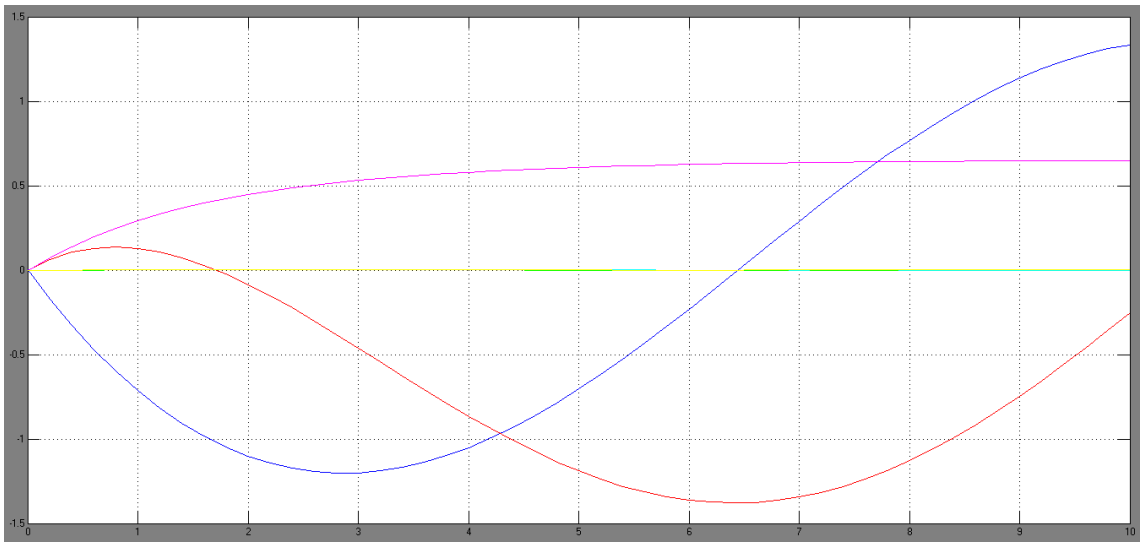

Figure 54 - Trajectory 2, SVD method

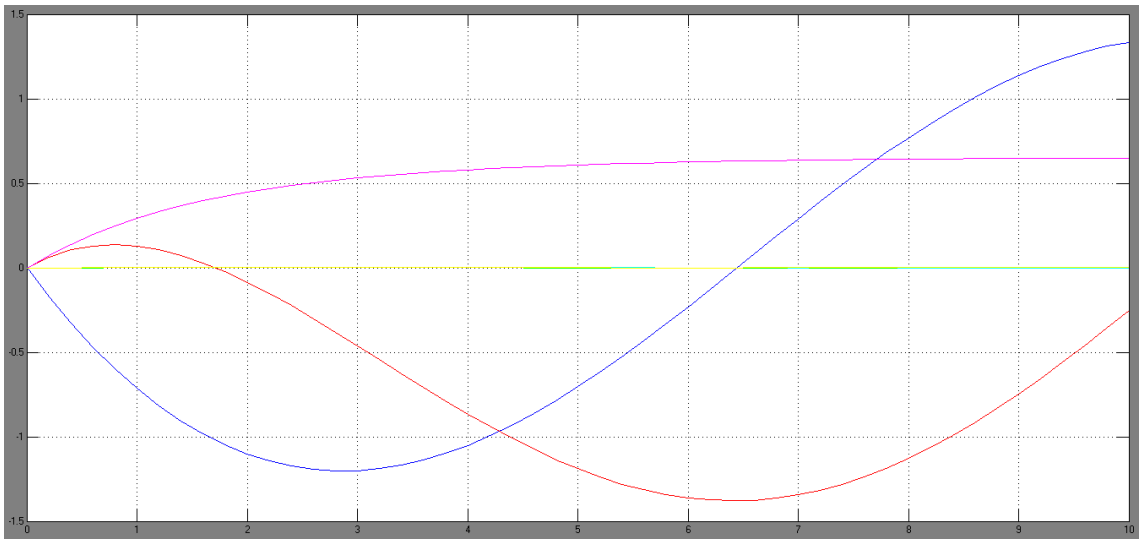

Figure 55 - Trajectory 2, pseudoinverse

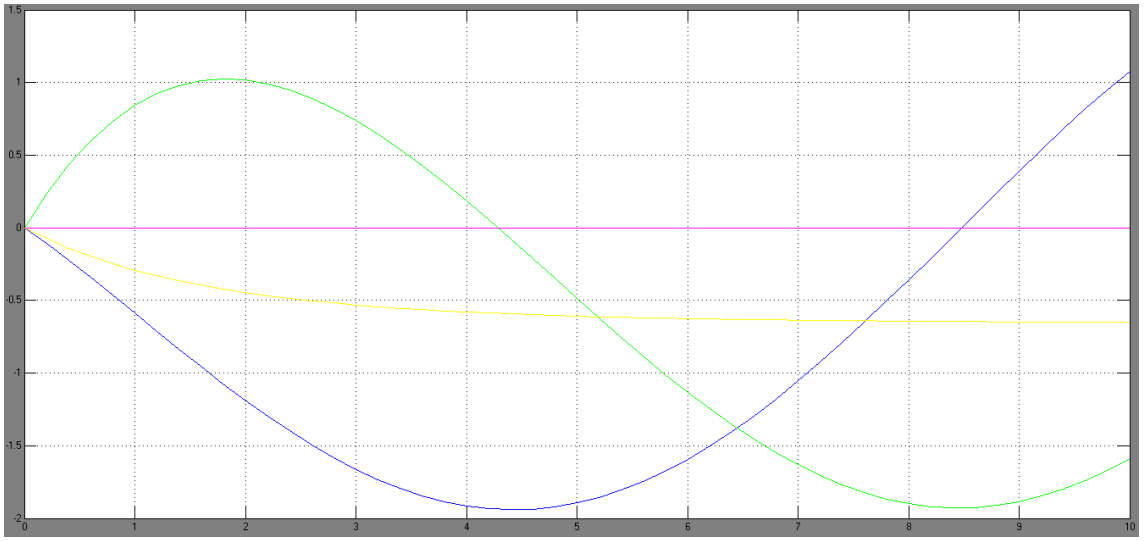

Figure 56 - Trajectory 3, DLS

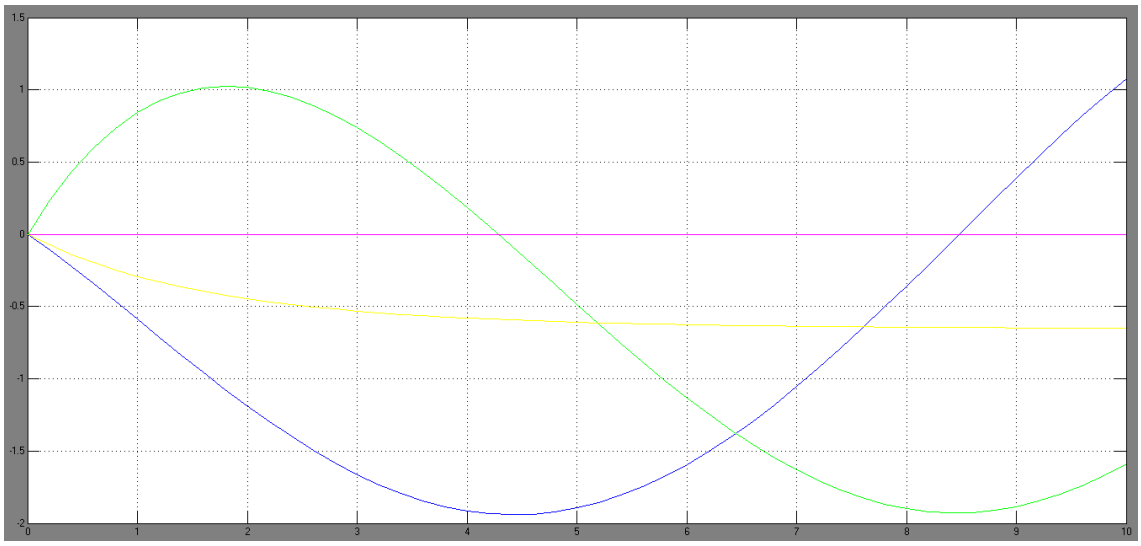

Figure 57 - Trajectory 3, SVD

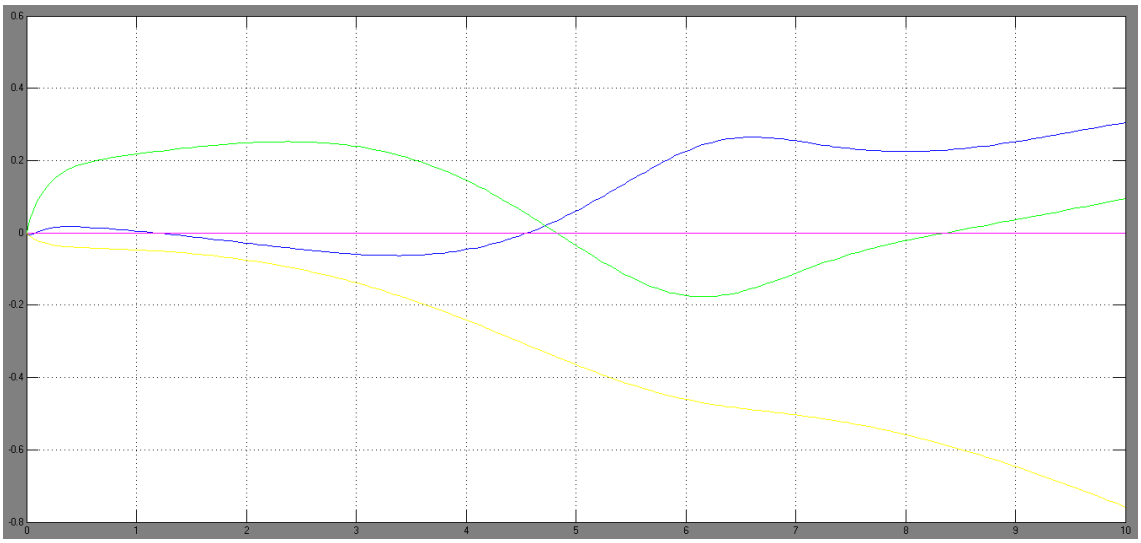

Figure 58 - Trajectory 3, transpose method

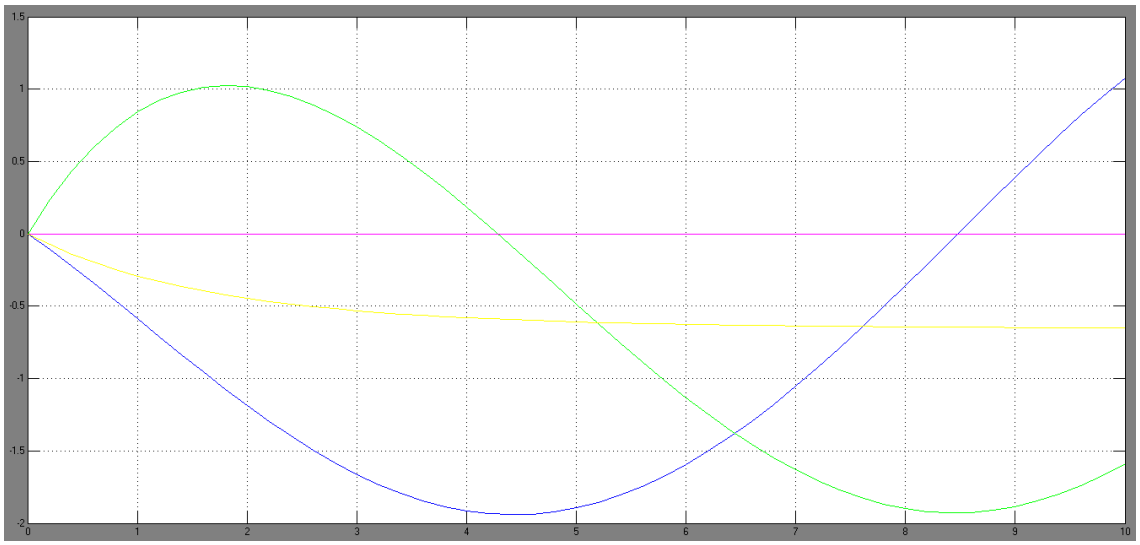

Figure 59 - Trajectory 3, pseudoinverse

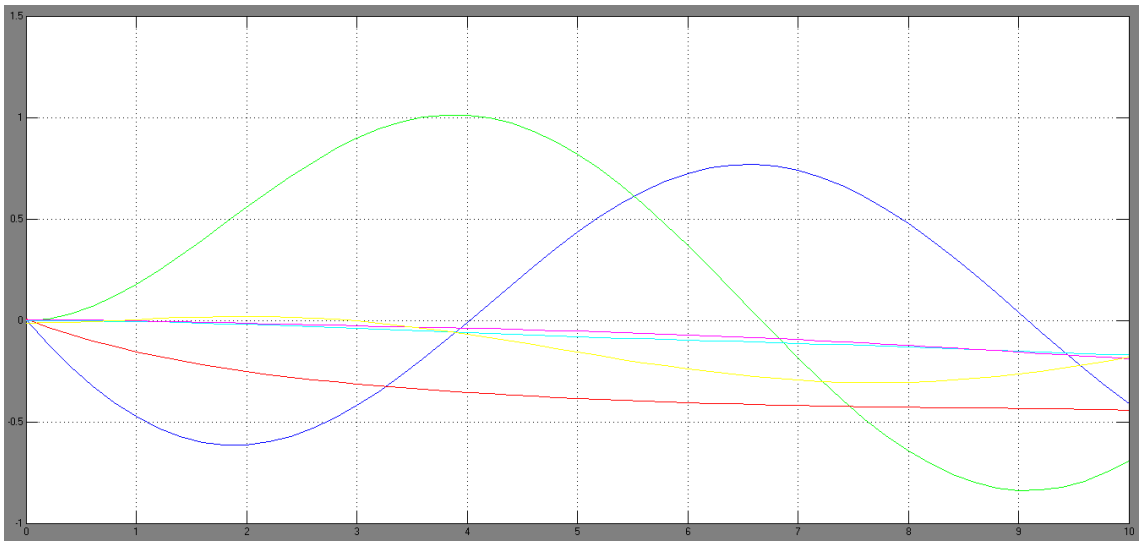

Figure 60 - Trajectory 4, DLS

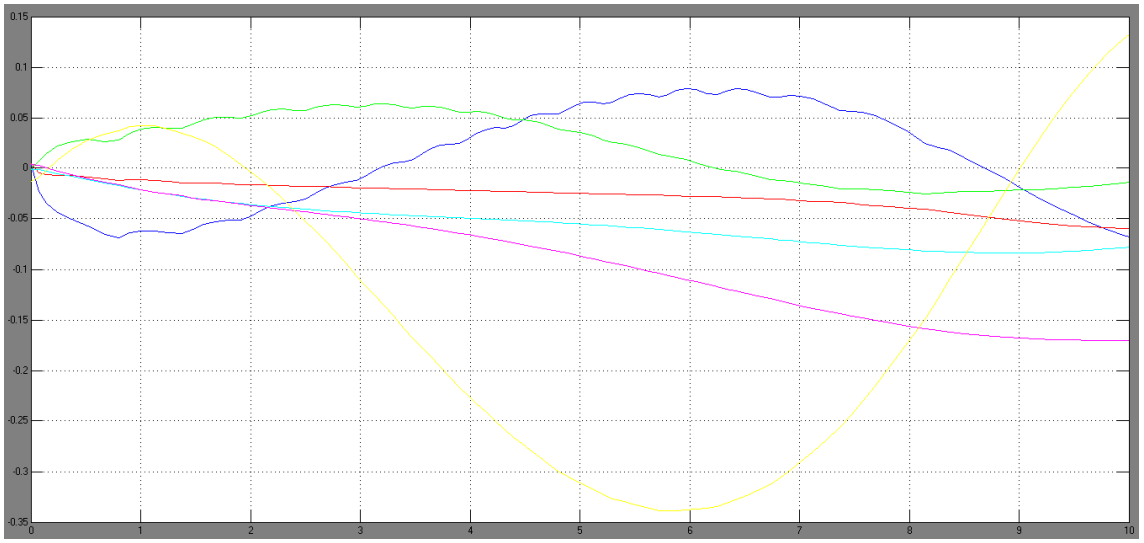

Figure 61 - Trajectory 4, Transpose

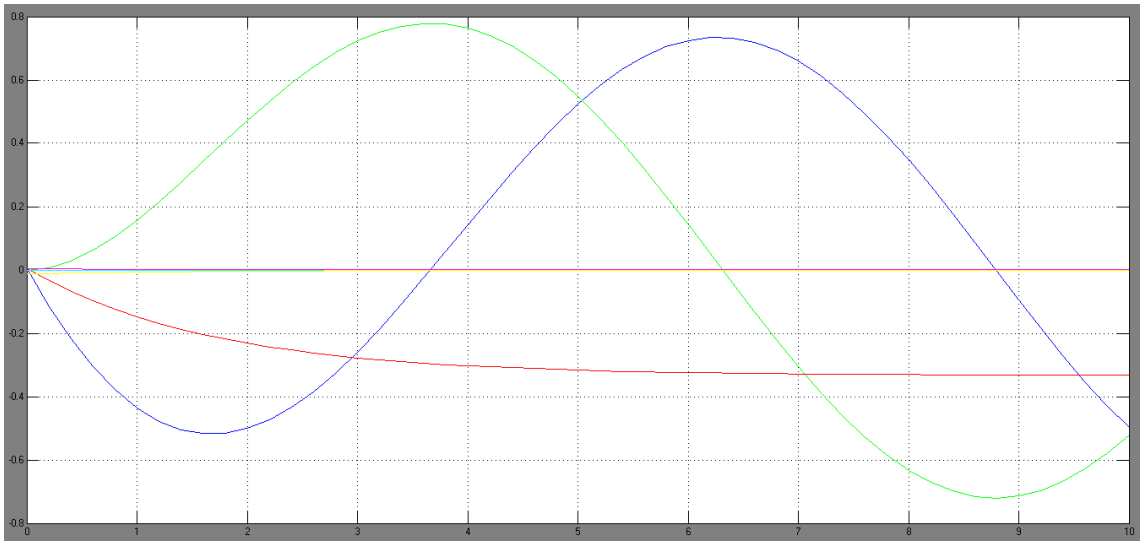

Figure 62 - Trajectory 4, SVD

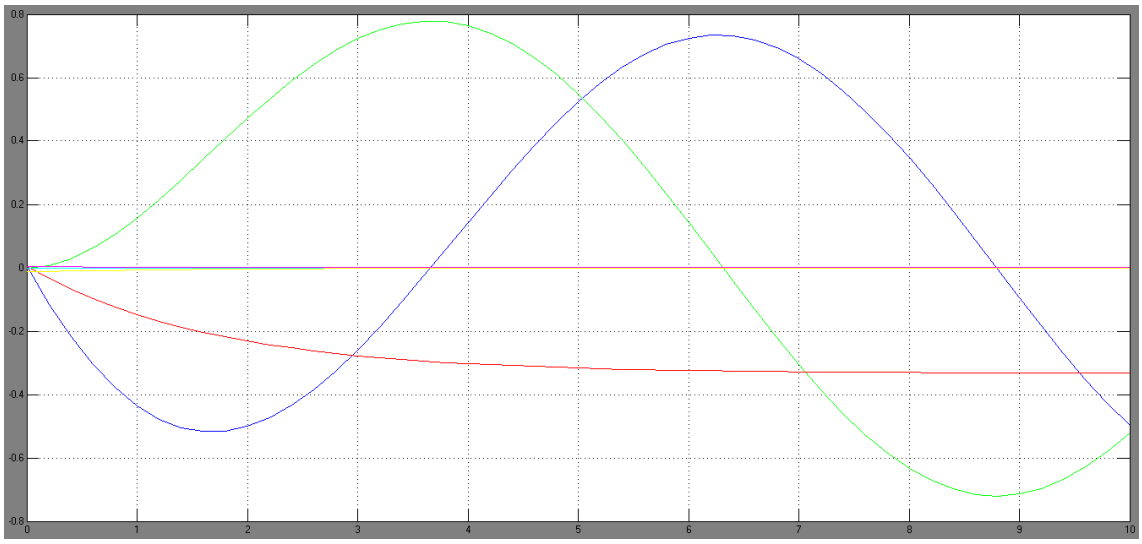

Figure 63 - Trajectory 4, pseudoinverse

5.4. Graphical joint space offset

In this section, it will be provided the difference between the real trajectory in the joint space and the projected one (obtained with the optimization algorithm). For all figures in this section, the horizontal axis corresponds to the time (in seconds) and the vertical axis corresponds to the following legend:

- Dark Blue = error for the 1st joint
- Green = error for the 2nd joint
- Red = error for the 3rd joint
- Cyan = error for the 4th joint
- Magenta = error for the 5th joint
- Yellow = error for the 6th joint

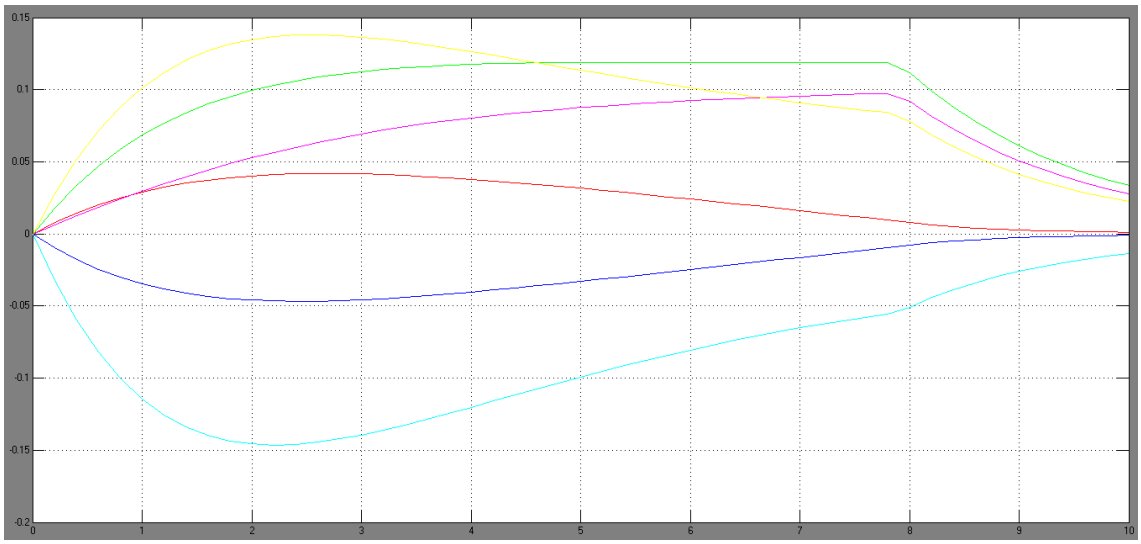

Figure 64 - Trajectory offset in trajectory 1, DLS

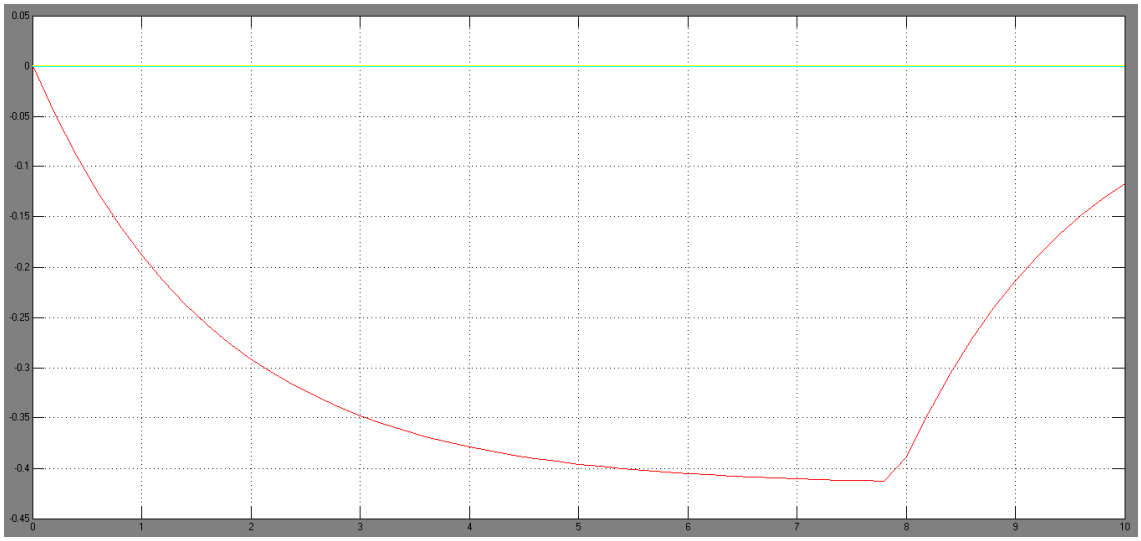

Figure 65 - Trajectory offset in trajectory 1, pseudoinverse

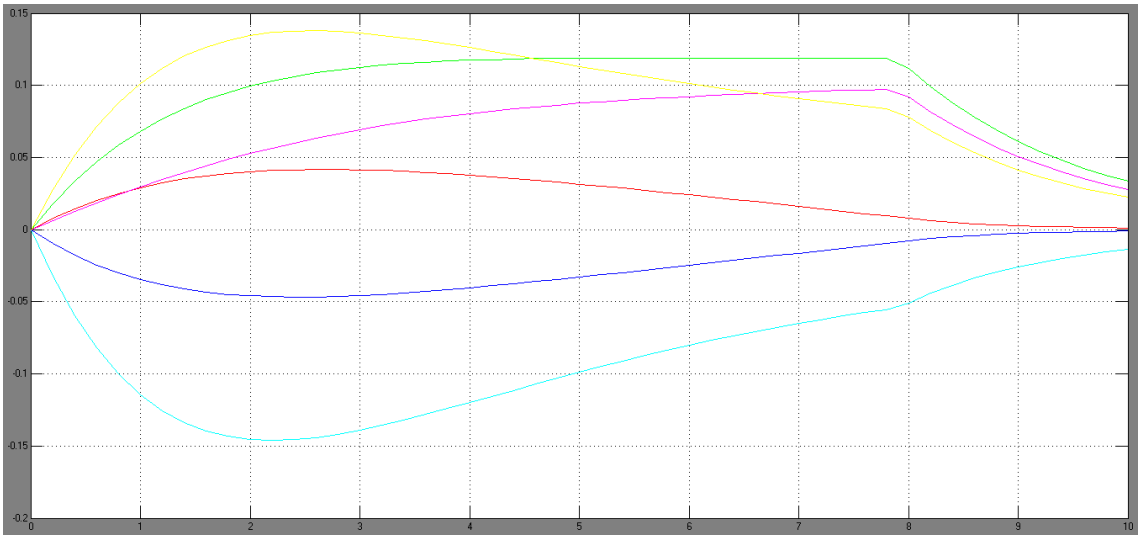

Figure 66 - Trajectory offset in trajectory 1, SVD

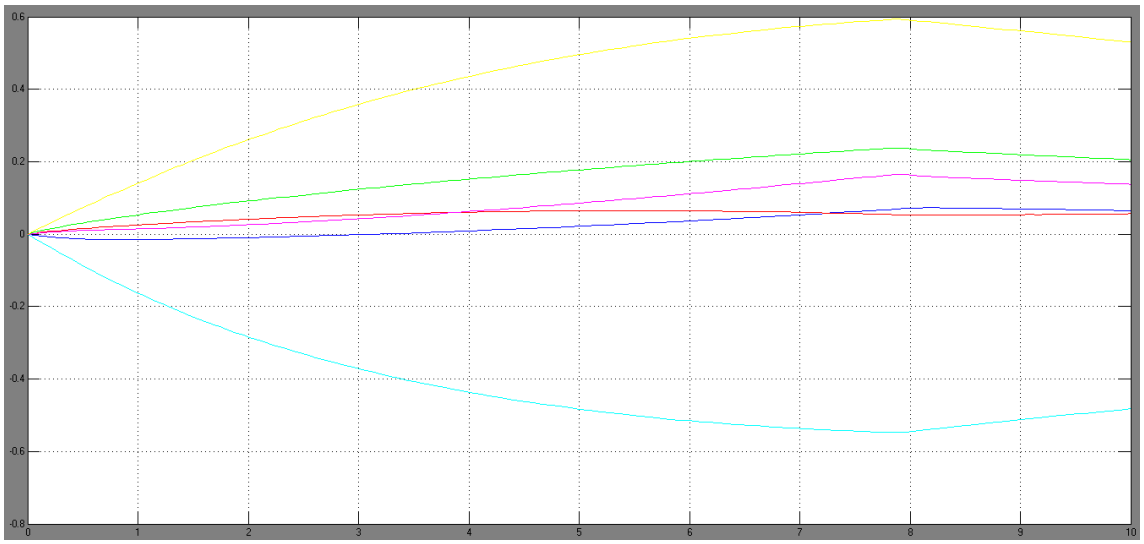

Figure 67 - Trajectory offset in trajectory 1, transpose method

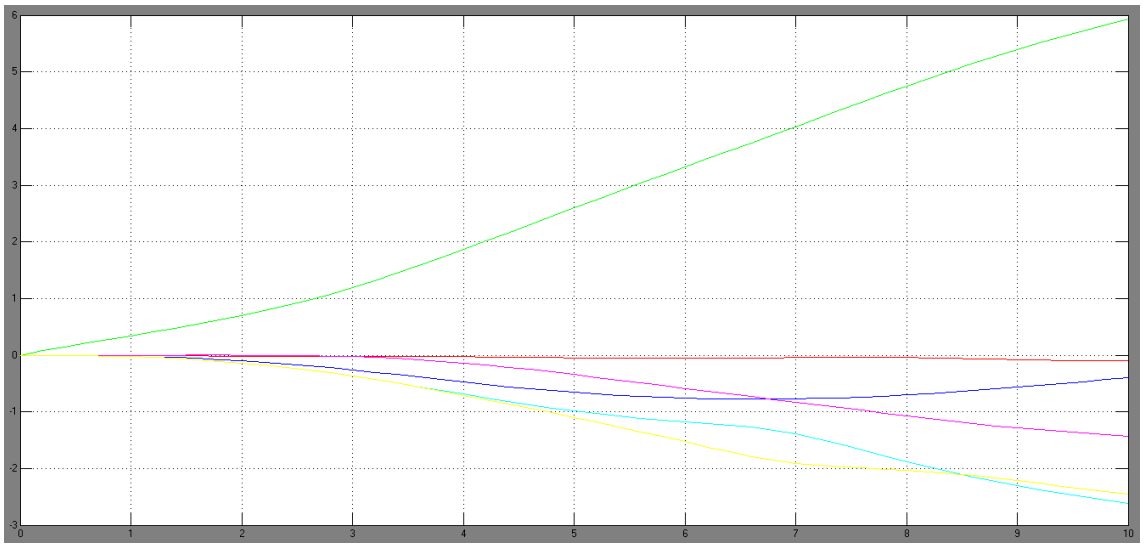

Figure 68 - Trajectory offset in trajectory 2, DLS

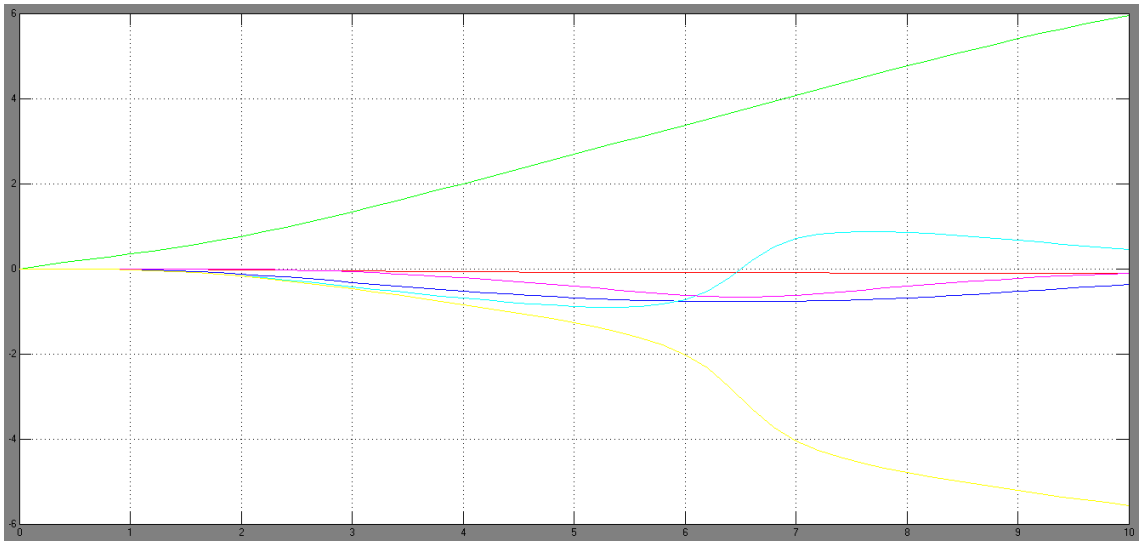

Figure 69 - Trajectory offset in trajectory 2, Pseudoinverse

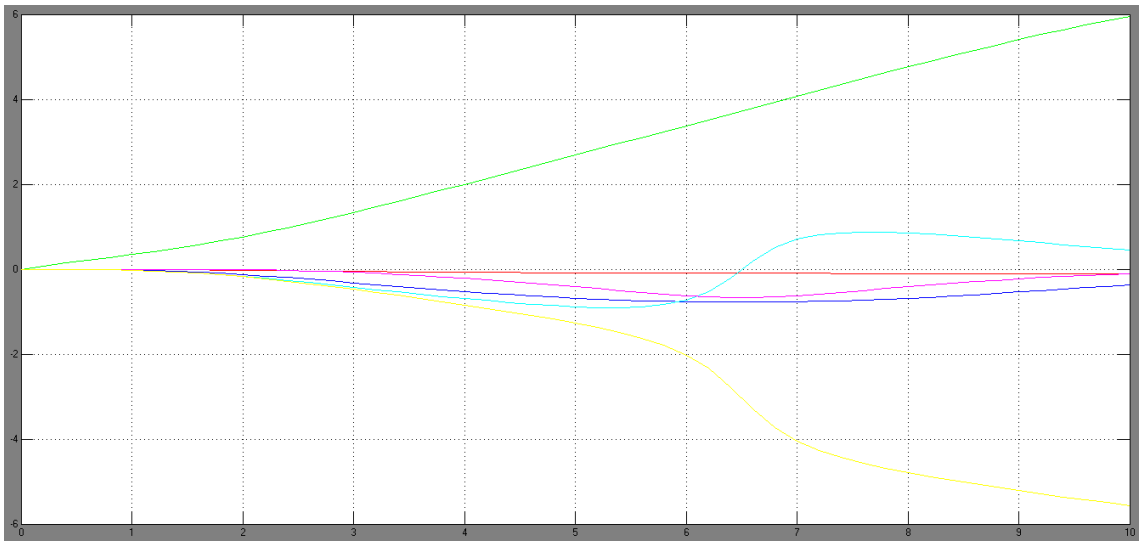

Figure 70 - Trajectory offset in trajectory 2, SVD

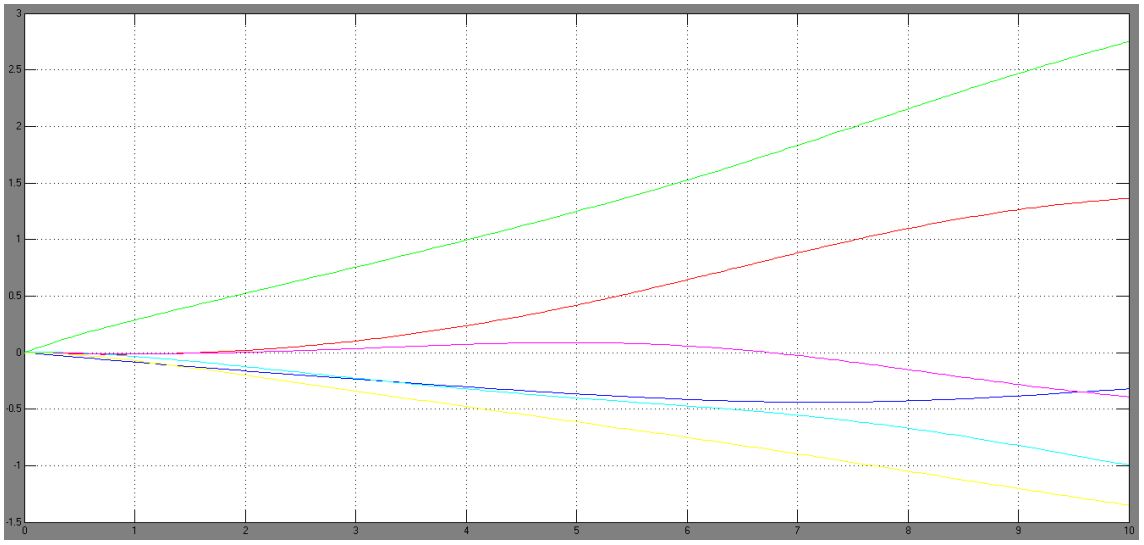

Figure 71 - Trajectory offset in trajectory 2, transpose

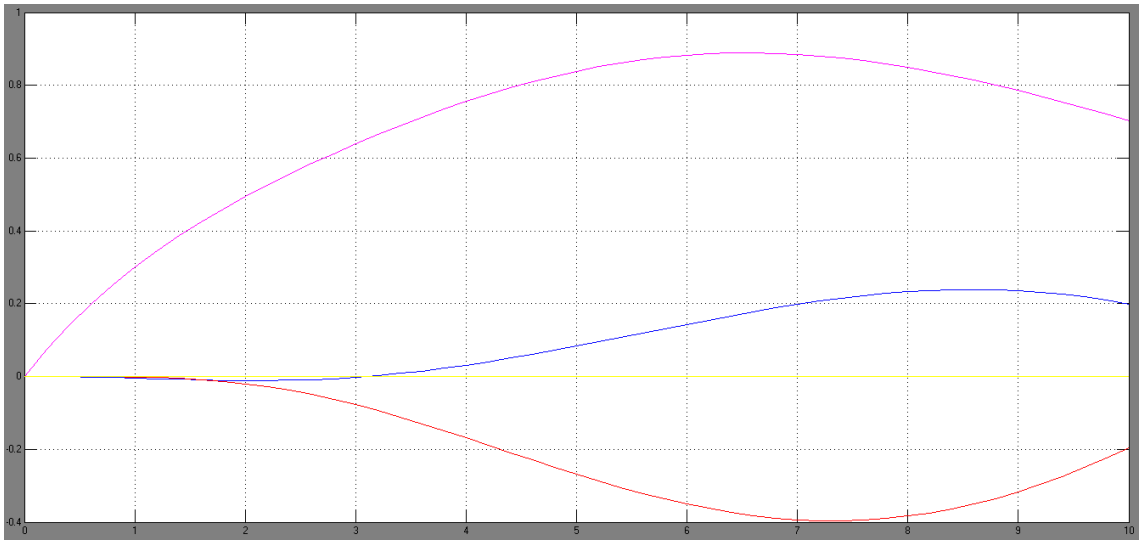

Figure 72 - Trajectory offset in trajectory 3, DLS

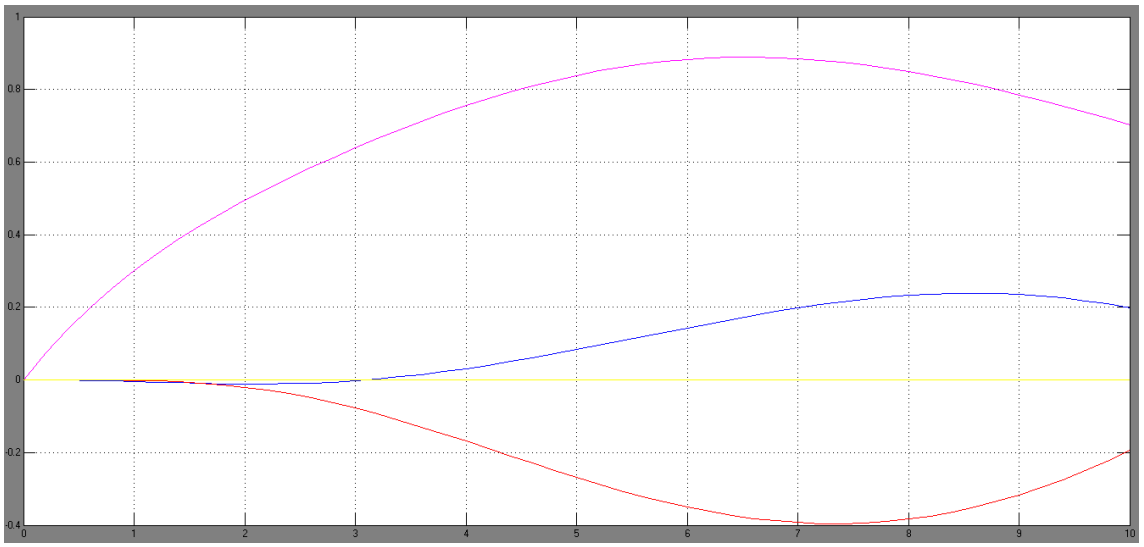

Figure 73 - Trajectory offset in trajectory 3, Pseudoinverse

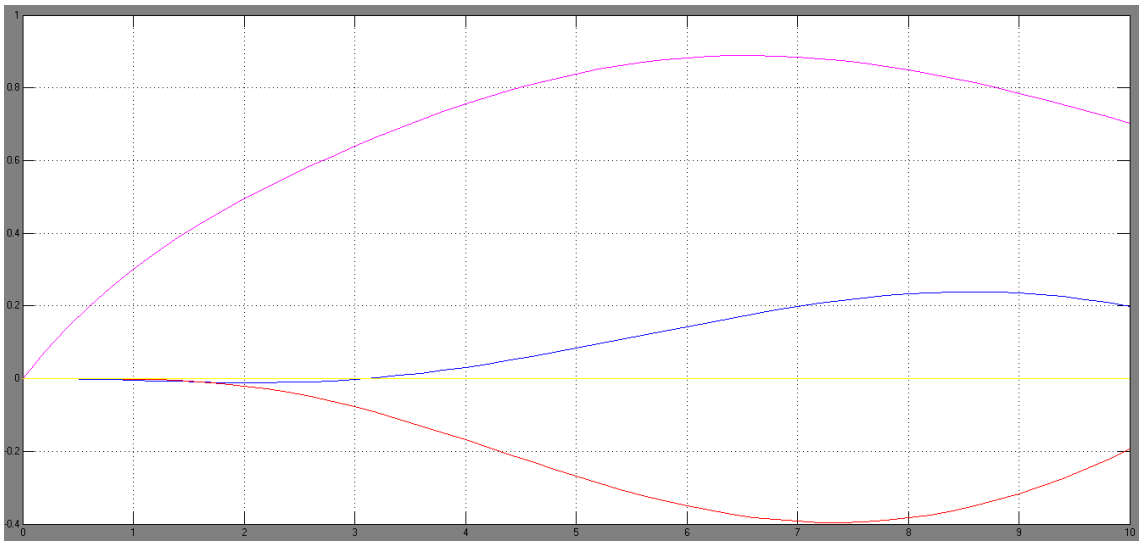

Figure 74 - Trajectory offset in trajectory 3, SVD

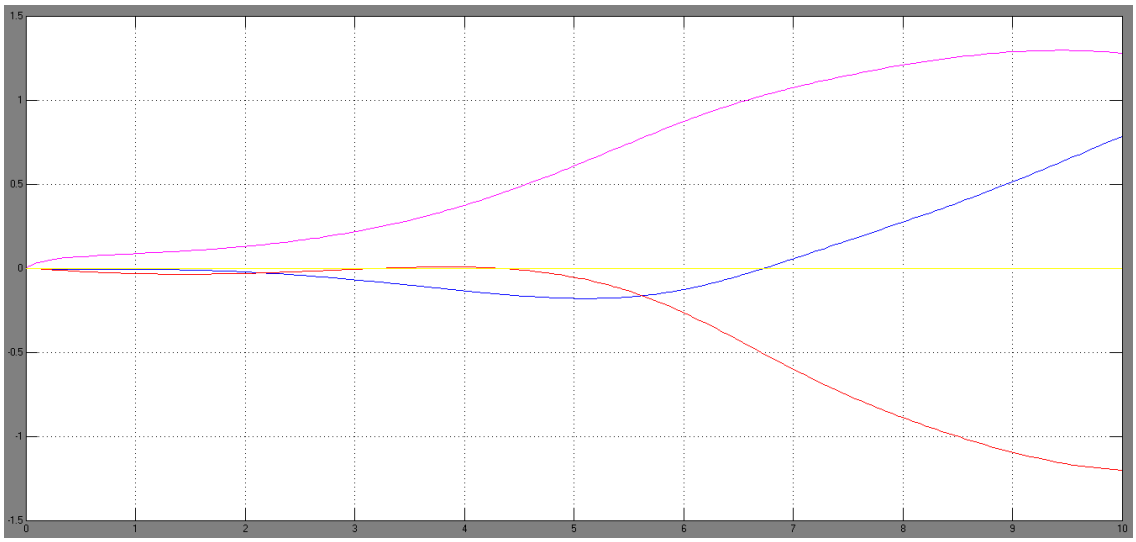

Figure 75 - Trajectory offset in trajectory 3, transpose

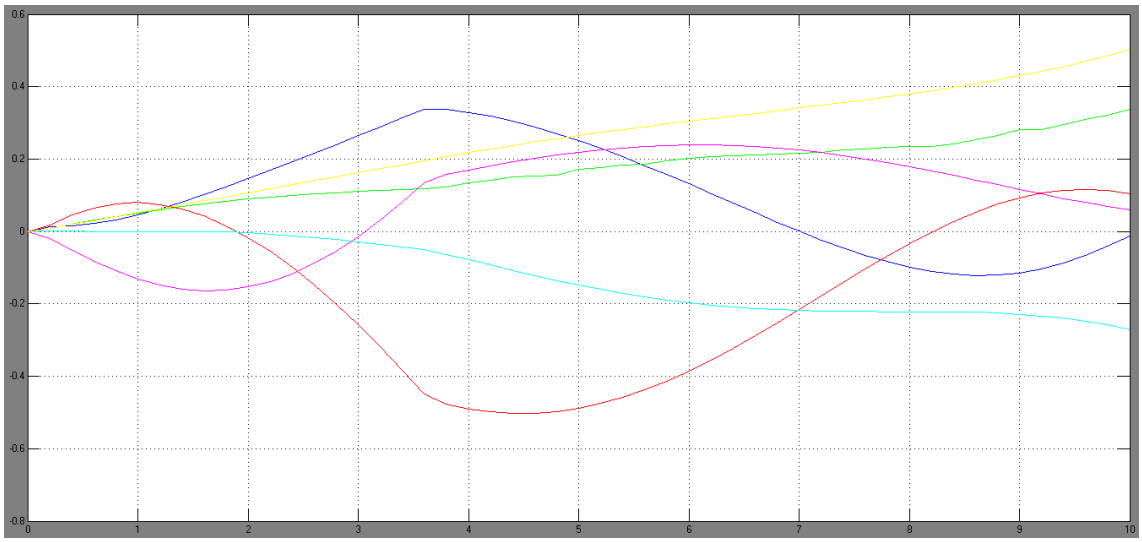

Figure 76 - Trajectory offset in trajectory 4, DLS

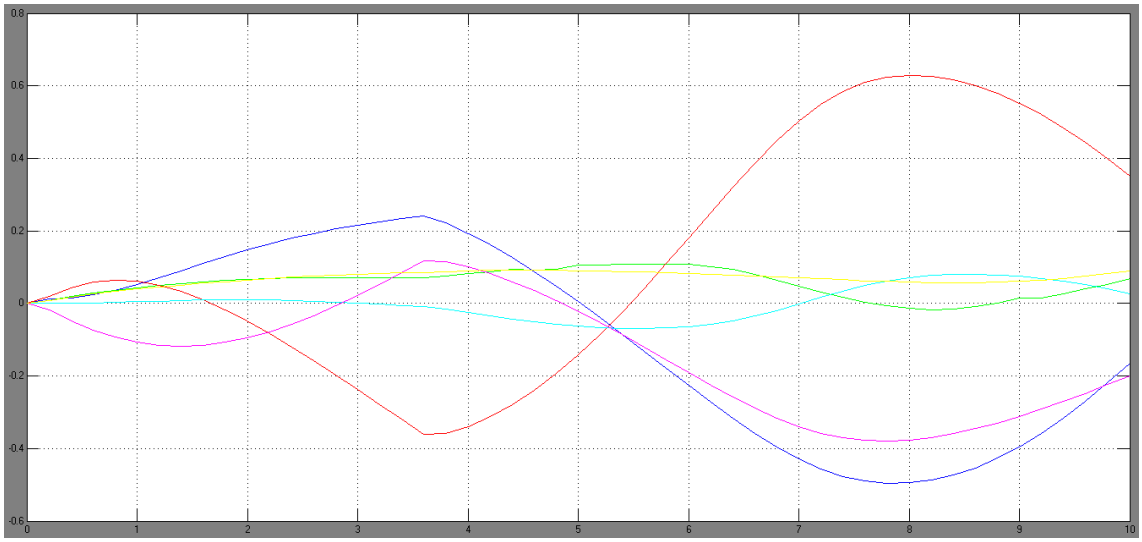

Figure 77 - Trajectory offset in trajectory 4, Pseudoinverse

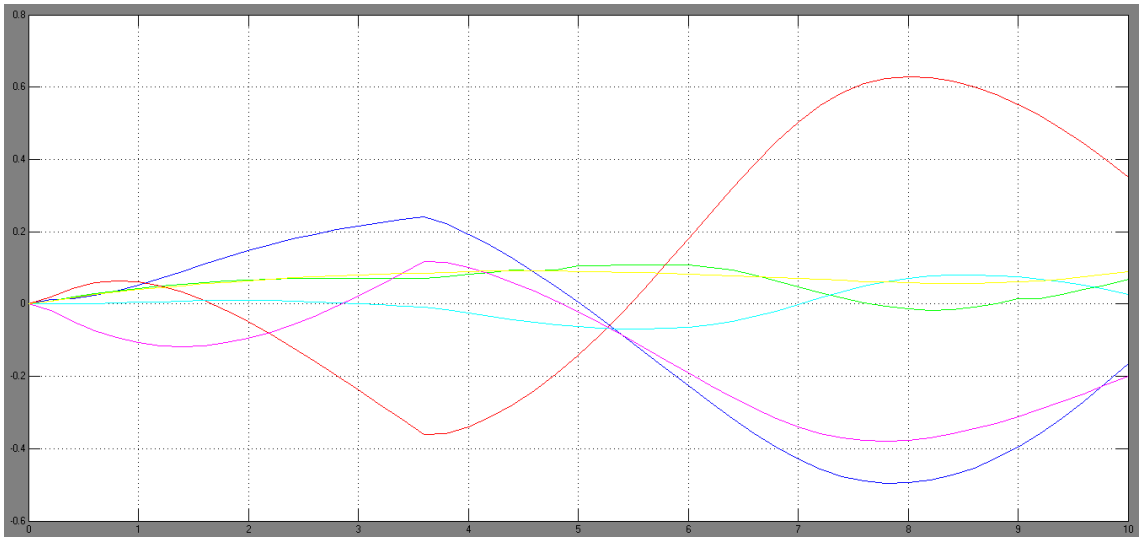

Figure 78 - Trajectory offset in trajectory 4, SVD

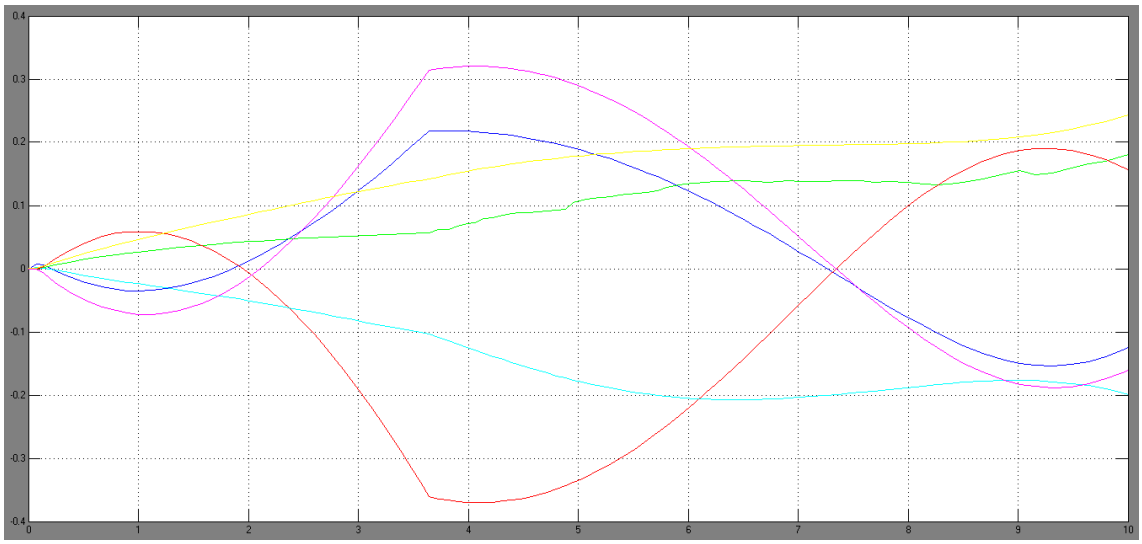

Figure 79 - Trajectory offset in trajectory 4, transpose

6. Discussion and Conclusion

The main objective of this work was to test kinematic control algorithms in real world surgery trajectories. The algorithm should well behave when in singular configuration either following the original trajectory in joint space or using other degrees of freedom to follow the trajectory in Cartesian space.

As analyzed in Section 3.2.3, the only singular configurations truly avoidable without changing position or orientation are the one caused by the 5^{th} joint angle equal zero. Some of those configurations have a symmetric non-singular configuration, and one may consider the motion between a quasi-singular configuration and its symmetric.

The first concern would be if the internal joint motion would affect the surrounding tissues. That should not be a problem since the operation site is inflated with carbon dioxide making enough room for the manipulator to move.

The second concern is about the small end-effector motion which is inevitable during the internal joint motion. In the worst case scenario, if the surgeon is performing a suture or other delicate procedure, will a 1 cm displacement affect his technique or the patient's outcome?

6.1. Trajectories

During the testing phase of this work, many different trajectories were considered in order to evaluate the performance of the kinematic control algorithms. The available computational tools for NOTESnail real-time 3-D visualization required a joint-space input, making it difficult to draw up trajectories based on intuition.

On the other hand, the old-fashion approach on drawing trajectories revealed important kinematic limitations on a 3-module NOTESnail design:

- If the approach while designing a trajectory was to maintain a fixed orientation while moving the end-effector's Cartesian

coordinates, the workspace became very limited. As shown in section 3.2.2, given the end-effector position and orientation, it is possible to retrieve the position of the 4^{th} and 5^{th} joints and the configuration is reachable if and only if:

- The position of the 4^{th} and 5^{th} joints is inside the 2-modules workspace shown in Figure 18;
- The 4^{th} and 5^{th} joint angles necessary to position the last module are inside the allowed range.

These two conditions make the number of possible trajectories with fixed orientation very limited when considering the size of the workspace.

- As shown in Figure 17, the [-3п/4;3п/4] range for the rotation joints combined with the [-п/2;п/2] range for the bending joint seems to be more than enough for building a half-sphere capable of reaching any desired point, but when considering these points in a trajectory, some problems emerge:
  - One should consider the suture motion, as shown in Figure 40, when done with common 4-DOF laparoscopic instruments. The circular motion done around the suture line when analyzed in joint variables (as shown in Figure 80) would give a continuous increasing value (from 0 to 2.n.п, where n is the number of suture rotations), which is not compatible with NOTESnail joint range limitations.

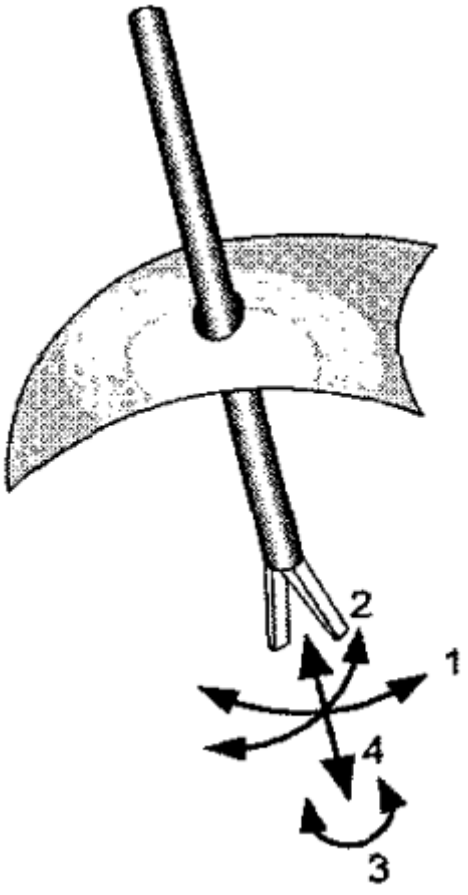

Figure 80 - Laparoscopic instrument joint variables

Comparing all inverse kinematics algorithms, a few considerations have to be pointed out:

6.2. Transpose method

The transpose method has the worse behavior among all solutions: it has a slow convergence and step instability (resulting in the comb-like graphic as shown in Figure 48, for example).

In the Cartesian space, the transpose method has one of the worse performances, but on the other hand, it keeps a closer distance to the expected joint space trajectory when considering a singularity-rich route.

On the other hand, following closer to the expected joint trajectory implies that there is no singularity avoidance using other degrees of freedom, which was one of the main goals of this work.

6.3. Damped least squares method

The damped least squares method has different performances in the Cartesian space and in the joint space according to the damping constant:

- In the real world reachable trajectories, DLS method has the same great performance as the pseudoinverse and SVD methods (both in Cartesian and joint space). In the case of a real world unreachable trajectory, the DLS has a better performance than pseudoinverse and SVD (both in Cartesian and joint space).
- In extreme singularities conditions, DLS method has an equivalent performance to SVD and pseudoinverse methods when analyzing the Cartesian space and the best performance when analyzing the joint space offset.
- The damping constant affects the overall performance when varying between the values of [-1;1]. Out of this range, the performance is kept saturated in the extreme values. The resulting performance x damping constant graphic has a Gaussian-like format centered in 0, as shown in .The best damping constant value found during the test is -0.01

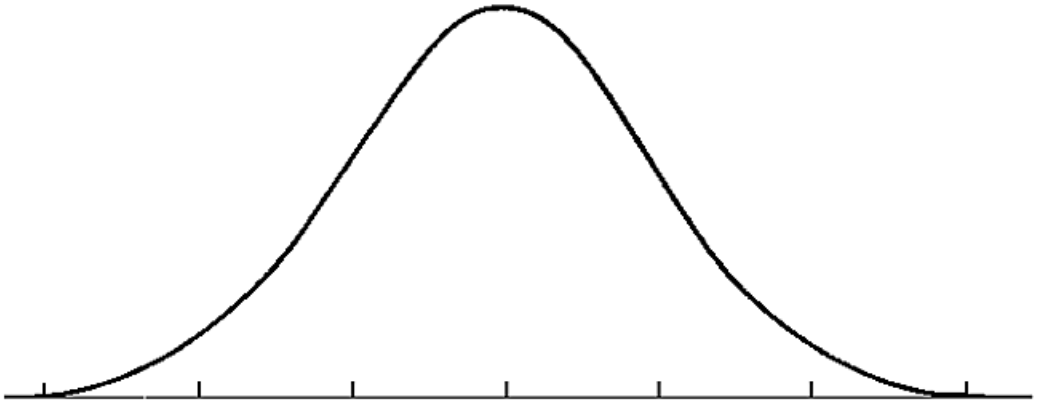

Figure 81 - damping constant x overall performance

6.4. Singular Value Decomposition

Similarly to the pseudoinverse method, the SVD method is very robust and stable algorithm. Both methods have a closer performance to the initial objective of this work: to use other degrees of freedom for singularity avoidance while keeping the desired trajectory in the Cartesian space. No other considerations are worth pointing out.

6.5. Pseudoinverse method

The pseudoinverse method, as the SVD method, has the behavior which is closest to this work's objective. Both methods have exactly the same performance when the null space projection vector is zero, but it points out the advantage of the pseudoinverse over SVD method.

By manipulating the null space projection vector, one can achieve to avoid singular configurations by creating internal motion, which is the great objective of this work. The best manipulation of the null space projection can be an objective of future work. The only manipulation proposed is shown in Equation 14:

$$null_{space} = [0\ , \frac{\frac{q_2}{\|q_2\|}}{\frac{\pi}{2} - \|q_2\|}\ , \frac{1}{q_3}\ , 0\ , \frac{1}{q_5}\ , 0]$$

Equation 14

This manipulator was though to keep the manipulator as far as possible of singular configuration. The problem with that algorithm is that those configurations become unreachable and a large portion of the workspace is lost.

6.6. Optimization algorithm

The optimization algorithm is very successful when planning trajectories and executing the inverse kinematics, but becomes impracticable when executing in real-time. Simulink simulations could not progress any greater than zero time-stamp and would also block the operating system.

6.7. Final considerations

Given all considerations about singular configurations in the NOTESnail system and the behavior of the inverse kinematics algorithms, the author of this work suggests that the singularity avoidance would benefit from increasing the number of construction modules.

The increase in the number of modules will provide the system with 8 degrees of freedom, 2 more that the necessary to reach any point with any orientation, what will provide any inverse kinematics algorithm the necessary redundancy to achieve the initial objective of singularity avoidance.

Given the collected data, the best option for the kinematic control is the pseudoinverse method. This algorithm answers to the main goal which was to use other degrees of freedom to overcome a rank-deficient Jacobian matrix simple inverse, as well as behaves perfectly along singularity-free paths.

The possibility of exploring null space projection vectors gives the pseudoinverse method an advantage over the Singular Value Decomposition and can be a powerful tool when the manipulator is forced into a border singularity. The tracking error in the Cartesian space is acceptable, especially because there are no computer-generated paths, but a human visual control.

Finally, this work showed to importance of analyzing the physical meaning of each singularity and which kinematic control algorithm is more appropriated for compensating a 3-module mechanical design the is very suited for the propos of NOTES but highly limited in terms of possible trajectories.

7. Bibliography

Abo-Hammour, Z., Mirza, N., Mirza, S., & Arif, M. (31 de December de 2002). Cartesian path generation of robot manipulators using continuous genetic algorithms. *Robotics and Autonomous Systems, 41*(4), pp. 179-223.

American College of Surgeons. (2007). *A Guide to Surgical Specialists*. Acesso em 26 de 10 de 2011, disponível em American College of Surgeons: http://www.facs.org/public_info/yourhealth/guide.html

American Society of Colon & Rectal Surgeons. (2008). *Laparoscopic Surgery - What Is It?* Acesso em 26 de 10 de 2011, disponível em Patients & Public Information: http://www.fascrs.org/patients/treatments_and_screenings/laparoscopic_surgery/

Balestrino, A., de Maria, G., & Sciavicco, L. (1984). Robust Control of Robotic Manipulators. *Proceedings of the 9th IFAC World Congress*, *5*, pp. 2435-2440.

Bowman, D. E. (02 de 2006). ASGE/SAGES Working Group on Natural Orifice Translumenal Endoscopic Surgery: White Paper October 2005. *Gastrointestinal Endoscopy, 63*(2), pp. 199-203.

Buss, S. R. (2004). *Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares methods.* University of California, Department of Mathematics, San Diego.

Capasso, L. (2002). *Principi di storia della patologia umana : corso di storia della medicina per gli studenti della Facoltà di medicina e chirurgia e della Facoltà di scienze infermieristiche.* Roma: SEU.

Cerveri, P. (2008). Strumenti micro robotici modulari innovativi per la chirurgia transluminale endoscopica. *Progetto di Ricerca*. Milan, MI, Italy.

Corke, P. (December de 2008). *Robotics Toolbox for Matlab.* Acesso em 15 de 11 de 2011, disponível em Peter Corke: http://www.petercorke.com

Geoffrey, B., Timothy, A., Jeffrey, C., Mihir, D., Edward, C., & Ralph, C. (11 de 2008). Nomenclature of Natural Orifice Translumenal Endoscopic Surgery (NOTES) and Laparoendoscopic Single-Site Surgery (LESS) Procedures in Urology. *Journal of Endourology, 22*(11), pp. 2575-2582.

Girard, M., & Maciejewski, A. A. (1985). Computational modeling fot he computer animation of legged figures. *Computer Graphics, 19*, pp. 263-270.

Golub, G., & Reinsch, C. (29 de 04 de 1970). Singular value decomposition and least squares solutions. *Numerische Mathematik, 14*(5), pp. 403-420.

Haber, G.-P., Crouzet, S., Kamoi, K., Berger, A., Aron, M., Goel, R., et al. (June de 2008). Robotic NOTES (Natural Orifice Translumenal Endoscopic Surgery) in Reconstructive Urology: Initial Laboratory Experience. *Urology, 71*(6), pp. 996-1000.

Hess-Coelho, T. (2004). Modelagem cinemática 3-D. São Paulo, SP, Brasil.

Hutchinson, S., Hager, G. D., & Corke, P. I. (October de 1996). A tutorial on visual servo control. *IEEE Transactions on robotics and automation, 12*(5), pp. 651-67.

Intuitive Surgical. (2010). *The da Vinci Surgical System*. Acesso em 26 de 10 de 2011, disponível em Intuitive Surgical: http://www.intuitivesurgical.com/products/davinci_surgical_system/

Kalloo, A. N., Singh, V. K., Jagannath, S. B., Niiyama, H., Hill, S. L., & Vaughn, C. A. (07 de 2004). Flexible transgastric peritoneoscopy: a novel approach to diagnostic and therapeutic interventions in the peritoneal cavity. *Gastrointestinal Endoscopy, 60*(1), pp. 114-117.

Loulmet, D., Carpentier, A., d'Attellis, N., Berrebi, A., Cardon, C., Ponzio, O., et al. (1999). Endoscopic coronary artery bypass grafting with the aid of robotic assisted instruments. *Journal of Thorac Cardiovasc Surgery*, pp. 118:4-10.

Maciejewski, A. A., & Klein, C. A. (1985). Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *International Journal of Robotic Research, 4*, pp. 109-117.

Misra, P., Patel, R. V., & Balafoutis, C. A. (1988). Robust Control of Robot Manipulators in Cartesian Space. *American Control Conference*, (pp. 1351-1356).

Oemoto, D., & Ang Jr., M. (12 de September de 2007). Singularity robust algorithm in serial manipulators. *Robotics and Computer-Integrated Manufacturing, 25*, pp. 122-134.

Ogata, K. (2001). *Modern Control Engineering* (4 ed.). Prentice Hall.

Oleynikov, D. (10 de 2008). Robotic Surgery. *Surgical Clinics of North America, 88*(5), pp. 1121-1130.

Oyama, E., Chong, N. Y., Agah, A., & Maeda, T. (2001). Inverse kinematics learning by modular architecture neural networks with performance prediction networks. *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, *1*, pp. 1006-1012.

Pai, R., Fong, D., Bundga, M., Odze, R., Rattner, D., & Thompson, C. (Setembro de 2006). Transcolonic endoscopic cholecystectomy: a NOTES survival study in a porcine model. *Gastrointestinal Endoscopy, 64*(3), pp. 428-434.

Paul, R. P. (1981). *Robot Manipulators: Mathematics, Programming, and Control.* Cambridge, MA, USA: The MIT Press.

Rao, G., Reddy, N., & Banerjee, R. (04 de 2008). NOTES: Human Experience. *Gastrointestinal Endoscopy Clinics of North America, 18*(2), pp. 361-370.

Rieder, E., & Swanstrom, L. (September de 2011). Advances in cancer surgery: Natural orifice surgery (NOTES) for oncological diseases. *Surgical Oncology, 20*(3), pp. 211-218.

Roan, S. (2005). *Anesthesia's effects may linger after patient goes home*. Acesso em 25 de 10 de 2011, disponível em Pittsburgh Post-Gazette: http://www.post-gazette.com/pg/05201/540318.stm

Sanderson, A. C., & Weiss, L. E. (1980). Image-based visual servo control using relational graph error signals. *Proc. IEEE*, pp. 1074-1077.

Spong, M., Hutchinson, S., & Hutchinson, M. (2004). *Robot Dynamics and Control* (1 ed.). Wiley.

Tolani, D., Goswami, A., & Badler, N. (2000). Real-Time Inverse Kinematics Techniques for Anthropomorphic Limbs. *Graphical Models, 62*, pp. 353-388.

Velanovich, V. (01 de 01 de 2000). Laparoscopic vs open surgery. *Surgical Endoscopy*, pp. 16-21.

Wampler, C. W. (1988). Applications of damped least-squares methods to resolved-rate and resolved-acceleration control of manipulators. *Journal of Dynamic Systems, Measurement, and Control, 110*, pp. 31-38.

Wang, L.-C., & Chen, C. C. (August de 1991). A combined optimization method for solving the inverse kinematics problems of mechanical manipulators. *Robotics and Automation, IEEE Transactions on, 7*(4), pp. 489-499.

Whitney, D. E. (June de 1969). Resolved motion rate control of manipulators and human prostheses. *IEEE Transactions on Man-Machine Systems, 10*(2), pp. 47-53.

Yoshikawa, T. (1985). Dynamic manipulability of robot manipulators. *Journal of Robotic Systems, 2*, pp. 113-124.

8. Appendix A

8.1. Robot object building MATLAB code

```
clear L

d=3.5;

L{1} = link([pi/2  0 0 0],'standard');
L{2} = link([-pi/2 0 0 d],'standard');
L{3} = link([pi/2  0 0 0],'standard');
L{4} = link([-pi/2 0 0 d],'standard');
L{5} = link([pi/2  0 0 0],'standard');
L{6} = link([0     0 0 d],'standard');

L{1}.qlim = [-pi/2  pi/2];
L{2}.qlim = [-pi    pi];
L{3}.qlim = [-pi/2  pi/2];
L{4}.qlim = [-pi    pi];
L{5}.qlim = [-pi/2  pi/2];
L{6}.qlim = [-pi    pi];

L{1}.m = 0.033;
L{2}.m = 0.015;
L{3}.m = 0.033;
L{4}.m = 0.015;
L{5}.m = 0.033;
L{6}.m = 0.015+0.030;  % with microcamera

% Center of Mass in respect to the origin of each link
L{1}.r = [ 0.000179    0.000243     0.025109 ];
L{2}.r = [ 0           0.017786     0.000012 ];
L{3}.r = [ 0.000179    0.000243     0.025109 ];
L{4}.r = [ 0           0.017786     0.000012 ];
L{5}.r = [ 0.000179    0.000243     0.025109 ];
L{6}.r = [ 0           0.017786-0.0177     0.000012 ];

% Momento of inertia (from CAD)
L{1}.I = [  7.543e-006   2.337e-006      8.587e-006     2.147e-6   -
3.194e-7      8.8e-7];
L{2}.I = [  9.645e-007   1.696e-006      1.693e-006     1.44e-7    -
1.311e-7     -1.522e-7];
L{3}.I = [  7.543e-006   2.337e-006      8.587e-006     2.147e-6   -
3.194e-7      8.8e-7];
L{4}.I = [  9.645e-007   1.696e-006      1.693e-006     1.44e-7    -
1.311e-7     -1.522e-7];
L{5}.I = [  7.543e-006   2.337e-006      8.587e-006     2.147e-6   -
3.194e-7      8.8e-7];
L{6}.I = [  9.645e-007   1.696e-006      1.693e-006     1.44e-7    -
1.311e-7     -1.522e-7];

% Motor inertia (from datasheet)
L{1}.Jm =  9.5e-010;
L{2}.Jm =  9.5e-010;
L{3}.Jm =  9.5e-010;
L{4}.Jm =  9.5e-010;
L{5}.Jm =  9.5e-010;
L{6}.Jm =  9.5e-010;
```

```
% Motor reduction
L{1}.G =  1024*14/3;
L{2}.G =  1024*33/6;
L{3}.G =  1024*14/3;
L{4}.G =  1024*33/6;
L{5}.G =  1024*14/3;
L{6}.G =  1024*33/6;

% viscous friction (motor referenced) (from datasheet)
L{1}.B =   8e-7;
L{2}.B =   8e-7;
L{3}.B =   8e-7;
L{4}.B =   8e-7;
L{5}.B =   8e-7;
L{6}.B =   8e-7;


bot3 = robot(L);

bot3.name = '3rd module';

bot3.manuf = 'PoliMi';

clear L d;
```

8.2. Optimization algorithm in MATLAB code

```
function [qt yout] = ikinebot_full (robot,tr,qi)

liminf = [-pi/2 -3*pi/4 -pi/2 -3*pi/4 -pi/2 -3*pi/4];
limsup = [pi/2 3*pi/4 pi/2 3*pi/4 pi/2 3*pi/4];
f = @(x)full_matrix_op(x,robot,tr);
[qt yout] = fmincon(f,qi,[],[],[],[],liminf,limsup);

end

function y = full_matrix_op(x,robot,tr)

y = abs(norm(tr-fkine(robot,x)));

end
```

8.3. Singular Value Decomposition in MATLAB code

```
function y = svd_inv(J)

[U,S,V] = svd(J);

y = inv(U*S*V');

end
```

8.4. 2-module workspace Mesh in MATLAB code

```
d = 3.5;
n = 50;
% calota interna 1
theta = (-n:2:n)/n*pi/2;
phi = (-n:2:n)'/n*pi/2;
cosphi = cos(phi);
dsinphi = d*sin(phi);
dcostheta = d*cos(theta);
dsintheta = d*sin(theta);

x = cosphi*dcostheta + ones(n+1,1)*dsintheta;
y = cosphi*dsintheta - ones(n+1,1)*dcostheta;
z = dsinphi*ones(1,n+1);
c = ones(n+1);

surf(x,y,z,c)

hold on

% calota interna 2
theta = (-n:2:n)/n*pi/2;
phi = (-n:2:n)'/n*pi/2;
cosphi = cos(phi);
dsinphi = d*sin(phi);
dcostheta = d*cos(theta);
dsintheta = d*sin(theta);

x = -cosphi*dcostheta + ones(n+1,1)*dsintheta;
y = -cosphi*dsintheta - ones(n+1,1)*dcostheta;
z = -dsinphi*ones(1,n+1);
c = ones(n+1);

surf(x,y,z,c)


%calota externa
th1 = (-n:2:n)/n*pi/2;
th2 = (-n:2:n)'/n*pi/2;
costh2 = cos(th2);
dsinth2 = d*sin(th2);
dcosth1 = d*cos(th1);
dsinth1 = d*sin(th1);

xa = costh2*dsinth1 + ones(n+1,1)*dsinth1;
ya = - costh2*dcosth1 - ones(n+1,1)*dcosth1;
za = dsinth2*ones(1,n+1);
c = ones(n+1);

surf(xa,ya,za,c)

n = 20;

% borda 1
th2 = (-n:2:n)/n*pi/2;
th3 = (-n:1:0)'/n*pi/2;
dcosth3 = d*cos(th3);
```

```
dsinth3 = d*sin(th3);

x = -dcosth3*ones(1,n+1) -d*ones(n+1);
y = -dsinth3*cos(th2);
z = dsinth3*sin(th2);
c = ones(n+1);

surf(x,y,z,c)

% borda 2
th2 = (-n:2:n)/n*pi/2;
th3 = (0:1:n)'/n*pi/2;
dcosth3 = d*cos(th3);
dsinth3 = d*sin(th3);

x = dcosth3*ones(1,n+1) +d*ones(n+1);
y = dsinth3*cos(th2);
z = dsinth3*sin(th2);
c = ones(n+1);

surf(x,y,z,c)

x0 = 0;
y0 = -10.5;
z0 = 0;

scatter3(x0,y0,z0,50,'filled','y');

x0 = x0*ones(n+1);
y0 = y0*ones(n+1);
z0 = z0*ones(n+1);
r0 = 3.5;

% -pi <= theta <= pi is a row vector.
% -pi/2 <= phi <= pi/2 is a column vector.
theta = (-n:2:n)/n*pi;
phi = (-n:2:n)'/n*pi/2;
cosphi = cos(phi); cosphi(1) = 0; cosphi(n+1) = 0;
sintheta = sin(theta); sintheta(1) = 0; sintheta(n+1) = 0;

x = x0 + r0*cosphi*cos(theta);
y = y0 + r0*cosphi*sintheta;
z = z0 + r0*sin(phi)*ones(1,n+1);
c = 2*ones(n+1);

surf(x,y,z,c)

alpha(0.5)
axis equal
hold off
clear all
```