FINAL MASTER THESIS

INGEGNERIA DELL'AUTOMAZIONE

Visual control for fine grasping of deformable objects

Author: Serena Furlan
Director: Guillem Alenyà Ribas
Ponent: Cecilio Angulo Bahón
Call: January 2021

Universitat Politècnica de Catalunya

Escola Tècnica Superior d'Enginyeria Industrial de Barcelona

Institut de Robòtica i Informàtica Industrial

Abstract

This Master Thesis aims to investigate and assess different grasping strategies to allow a robot to grasp folded textiles. In general, the difficulty of the problem of grasping textiles is due to the variety of possible fabrics and patterns of the garments and to the high degree of achievable deformation shapes. The former implies complexity in perception, the latter causes difficulties in reconstructing the 3D model and predicting the behaviour of the objects.

This project focuses on the final approach motion that enables the grasping of the garment. Compared to the pre-grasp motion (first approach motion of the gripper to the garment), this is a challenging part that requires more precision from vision and more control accuracy. Moreover, the first part of the movement has already been developed in other projects that rely on hand-eye calibration and classical kinematics.

In particular, we are interested in providing the robot with a suitable strategy to grasp only the top layer of a folded garment, so we need a precision from vision that is not guaranteed by the camera on the robot head. Therefore we embedded an endoscopic camera in the robot hand in order to have a camera with a better point of view and mobile.

After having evaluated different techniques, like visual servoing, line detection and visual tracking, an approach based on line detection has been developed. This method is composed of a vision part followed by a control part. The vision phase exploits simple segmentation techniques like Canny edge detection and Hough transform in order to speed up the processing of the image and consequently the entire procedure. The control phase exploits the information coming from vision to elaborate new control messages sent to the Whole Body Controller (WBC) of the robot. These messages contain the new position in which we want to send the arm-tool link in order to approach carefully the garment.

Finally we have performed evaluation experiments using a TIAGo mobile manipulation robot in the Perception and Manipulation laboratory at IRI, a laboratory that simulates an apartment. Specifically, we have shown that despite the low precision of the WBC of the robot the closed-loop procedure designed works correctly with various types of folded garments, with an arbitrary number of layers, laying on different surfaces and under different lighting conditions.

Contents

1 Introduction

Robots have been in our lives for many years, but originally they were present only in industries, where they replaced humans in dangerous or tiring jobs without interacting with the workers. Instead, nowadays the robots are spreading in many environments, such as hospitals, e.g. Robot da Vinci for robot-assisted surgery [**1**], agriculture [**2**], search and rescue [**3**] and many others, included our houses, e.g. Roomba cleaning robots.

In particular, in recent years robotics has experienced a great progress in social and assistive domains. The goal of this field of robotics is to increase the well-being and the independence of people with disabilities or seniors, helping them to perform daily activities without the necessity of attendants. Since service robots have to interact with humans, this branch of research focuses on making the robots intrinsically safe to people, user-friendly and able to sense and take decisions in non-predefined environments [**4**].

Service robots should be able to execute many different tasks, one of the main tasks is object manipulation. Even if this task may be considered quite easy for humans, to make it performed by a robot, knowledge and techniques of many different fields (such as artificial intelligence, computer vision, machine learning) have to be combined.

Until now, researchers have been focused mostly on manipulation of rigid objects. However, in a domestic environment is very likely to deal with deformable objects (such as garments, cloths or bed-linen), that present many additional challenges with respect to rigid object manipulation, because of their unpredictable behaviour and variety of textures and shapes [**5**]. Moreover, studies concerning the grasp of garments concentrate on a rough grab of the entire cloth allowing collisions of the hand with the table, or grasping of garment edges and corners when they are clearly visible, for example, lying on a table. Up to our knowledge, the case of grasping a single layer of a folded garment with a general purpose robot hardware has not been tackled yet.

For these reasons, this project focuses on developing a technique to grasp with precision only the upper layer of a folded garment. The work has been developed as follows.

First of all, in Section 2 the state of the art in cloth manipulation is analyzed. In particular, we highlight the strategies more used to face the tasks of perception of the garment and detection of a grasping point, and we compare the most relevant visual servoing techniques currently used in the literature.

After that, Section 3 provides a description of the whole robotic system that will be used in the evaluation, including the main characteristics of TIAGo robot from the hardware and software point of view and the techniques exploited to perceive the garment and move to the pre-grasp position.

In Section 4 we evaluate some visual approaches and then we deeply explain the technique developed in this project both for the visual and the control phase.

In Section 5 first of all is provided the experimental validation of this procedure, highlighting how the approach has been improved facing to the real word constraints. Then we analyze the stability of the robot, of the controller, of vision and of the whole system. Finally we evaluate the average performances of the method, in terms of times.

After that, in Section 6 we estimate the total budget necessary for developing this project, taking into account electricity consumption, hardware, software and human resources.

Then in Section 7 we evaluate the social and environmental impact of the work, considering the electricity consumption (and consequently the amount of CO_2 emitted to produce it) and the production of the plastic gripper.

Finally in Section 8 we sum up the results of this project, assessing its strengths and proposing future improvements.

1.1 Objectives

The main objective of this project is to find a suitable strategy that allows a robot to grasp the top layer of folded garments, with an unspecified number of layers, when lying on a planar surface like a table.

Other projects that deal with the grasping of garments developed methods to grasp the entire garment, for example, by slipping the gripper between the garment and the table after a collision of the gripper with the table. Instead, in this project we want to accurately grasp only the last layer of garment avoiding collisions with the surface on which the garment is placed.

We assume that the pre-grasp motion has already been performed, so the problem is how to approach precisely to the top layer. To do this, we have to make small movements and close the loop to continuously correct the trajectory. To do so we use a camera integrated in the hand of the robot to have more accuracy with respect to the camera on the head of the robot. With this configuration we have also the advantage to have a subjective camera that moves in solidarity with the end–effector providing detailed views of the scene.

Then we want to evaluate the repeatability and the stability of the proposed strategy with different lighting conditions, garments, number of layers and types of surfaces.

1.2 Motivation

This work is under the scope of the European Research Council advanced grant CLOTHILDE (cloth manipulation learning from demonstration), a European project started in 2018.

The CLOTHILDE project aims to develop a theory of versatile cloth manipulation by robots based on task-oriented cloth representation, perception, probabilistic planning, learning from demonstration, failure diagnosis and informed requests for human help [**6**].

2 State of the art

In recent years the interest in service robots has increased, in particular a significant progress has been made in the development of techniques to manage robotic manipulation.

However, literature is more exhaustive with regard to the manipulation of rigid objects, because the manipulation of deformable objects presents many additional challenges.

Indeed deformable objects (such as clothes) have a variable shape, and usually a higher dimensional configuration space, that do not allow to apply the same manipulation techniques used for rigid objects.

Another issue is the absence of a general 3D model caused by the large variety of shapes, fabrics and design of the clothes that can be found in a domestic environment.

Then different fabric types have also diverse mechanical behaviours, so the grasp may be slightly different. These behaviours are quite hard to predict because fabrics have an anisotropic and nonlinear mechanical response.

In addition while 6 variables are sufficient to fully describe the state of a rigid object (3 variables for position and 3 for orientation), they are not enough to give a complete representation of the state of a deformable object, because manipulation produces deformations in the object.

Moreover, with rigid objects the grasp point is usually indifferent, while with deformable objects the grasp point affects the following configuration of the object, so for some tasks it is necessary to select a particular point and perform fine manipulation.

In this work we analyze different strategies to grasp a cloth, the entire process can be divided in 4 different phases (schematized in Figure 1):

- Corner detection;
- Movement pre-grasp, that can be performed applying classical kinematics laws after an hand-eye calibration;
- Final approach, that can be performed either with visual servoing techniques or line or object tracking;

- Grasp action;

Since the literature about deformable objects manipulation is not very complete, there is not a general technique to face the different phases, in the following we describe how similar tasks have been solved.

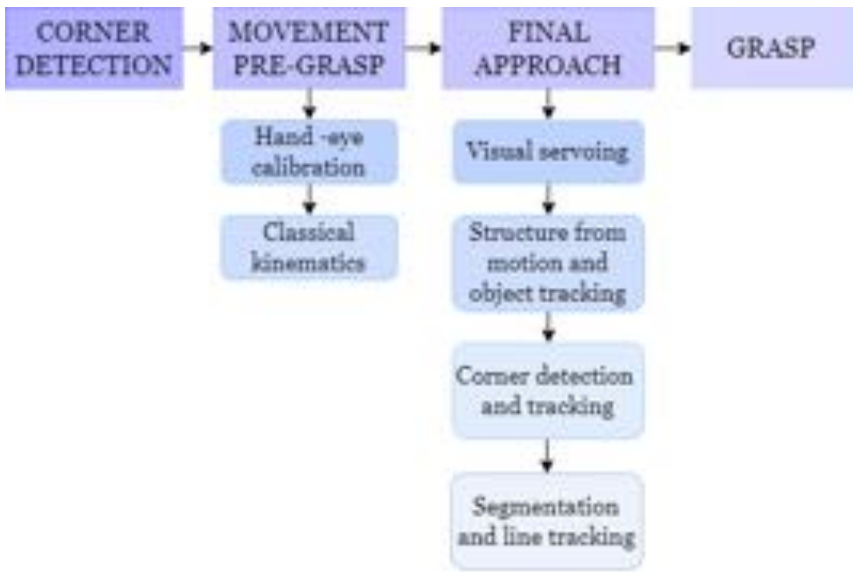

Figure 1: Diagram of the process

2.1 Perception of the garment and corner detection

The perception of the garment in the environment and the corner detection constitute the first part of our process. The initial configuration is a garment folded on a table.

In many studies the initial configuration is not a single garment, but a mass of clothes; an example is the work of K. Hamajima and M. Kakikura [7], that aims to isolate and unfold a garment from a washed mass. In this work first of all one cloth is identified among the others through a segmentation of the image based on colors. Then the grasping point is set in the middle of the cloth region, to avoid mistakes which may arise taking a point too close to the boundaries. After that the garment is hung up by holding the grasping point and two points on the edges of the garment are searched to perform a change of grasping. So the process of unfolding consist in a first grasp, followed by some rehandling to grasp the garment with two hands.

Similar assumptions can be found in [8], this works uses a dual arm robotic manipulator to unfold a garment. The garment is first autonomously grasped from a table by the robot and brought in front of it, then a rehandling is performed to detect points on the outline.

Edge points are searched because if a garment is helded by two points on the same edge, it can be brought to an approximately planar configuration, that would facilitate the following manipulations. The two outline points are detected using depths images.

Another approach [**9**] identifies the corners of the cloth as two exterior borders that cross at approximately a right angle in the surface of the garment. In particular they first identify the depth discontinuity edges of the image, with stereo or foreground/background segmentation. Using the map of edges classifications, they select candidate corner grasp points using a RANSAC-based algorithm that fits corners to the edge points.

For each of the 2-D corners found, a plane is fit (according to a Gaussian kernel centered on the tip) to the 3-D points corresponding to the image points contained in the corner. Finally the grasp point and orientation are chosen according to workspace constraints of the robot as well as to the desired grasp position and orientation relative to the corner tip.

P. Jiménez, in his work [**10**] assumes that the handling of cloth occurs via mechanical impactive grasps, so the fabric is hold by pressure between the gripper's fingers, performing a pinch grasp (acting on the same side) or a clamp grasp (acting on both sides of the cloth). Two families of grasp point determination methods are presented, one for generic and one for specific (i.e. with a precise localization) grasp points. Generic grasp point localization is region-oriented, the aim is to identify a region corresponding to the cloth and to take an arbitrary point inside it as grasp point.

A solution is to perform a segmentation of the image and to recover depth information from lateral cameras, stereovision or range cameras. An alternative is to exploit to some wrinkledness measure, because wrinkled areas provide good candidates for grasps.

Specific grasp point localization is feature-oriented, when the features are edges and corners they are extracted from outline information (which results either from a foreground–background segmentation or from edge detection methods).

2.2 Hand-eye calibration

Since TIAGo's camera is mounted on its head, before performing a movement it is fundamental to perform an hand-eye calibration, which means that it is needed to recover the relationship between the camera frame and the end-effector frame.

As reported in [**11**],[**12**], the hand-eye calibration is necessary for two types of tasks:

- Measure the position of an object into the robot workspace frame,
- Allow the robot to precisely move a sensor.

Our grasping task represents a case of the first category, indeed we want to grasp a garment whose pose in space is sensed through the camera of the robot, so we have to recover the relationship between the camera frame and the end-effector frame.

In particular we first recognize the object and determine its pose with respect to the camera, then we map the pose to the end-effector frame using a transformation between the camera and the Tool Center Point (TCP). After that the robot may move its end-effector towards the garment and grasp it.

This calibration tries to solve a homogeneous matrix equation of the form:

$$\mathbf{A} \cdot \mathbf{X} = \mathbf{X} \cdot \mathbf{B}$$

where

$$\mathbf{A} = \begin{pmatrix} \mathbf{R_A} & \mathbf{t_A} \\ \mathbf{0} & 1 \end{pmatrix}$$

is a 4×4 matrix that represents the transformation between two different positions of the sensor frame, $\mathbf{R_A}$ is a 3×3 matrix that represents the rotation and $\mathbf{t_A}$ is a 3-coordinates vector representing the position. $\mathbf{B}$ is a 4×4 matrix that represents the transformation between two positions of the end-effector frame and $\mathbf{X}$ is a 4×4 matrix representing the unknown transformation between the end-effector frame and the sensor frame.

Before obtaining the matrix $\mathbf{X}$ it is necessary to recover the intrinsic parameters of the camera, that are the focal length, the optical center and the skew coefficient.

Another formulation of this problem, proposed in [12] tries to solve the homogeneous matrix equation:

$$\mathbf{M} \cdot \mathbf{Y} = \mathbf{M}' \cdot \mathbf{Y} \cdot \mathbf{B}$$

where $\mathbf{M}$ and $\mathbf{M}'$ are perspective matrices associated with positions of the camera with respect to the calibration frame.

The advantage of this formulation is that the extrinsic and intrinsic parameters of the camera do not need to be made explicit.

2.3 Pose estimation

Pose estimation has been studied extensively for rigid objects, that present a fixed shape which allows to recover a model of the object.

An example is the work of M. Espinosa and S. Miller [13] that deals with the manipulation of cans. In this project the pose of each can is estimated detecting four coplanar points on its top and implementing an algorithm that requires as inputs: the camera parameters, the position of each point S_i in pixels and the nominal positions of the points S_i w.r.t. the origin of the can.

However this method is not applicable to our project because we do not have a geometric model of the configuration of the garment in space, so in general we cannot find 4 coplanar points.

A similar approach is to draw four black dots on the object, detect these blobs with some image processing techniques (extract the pixel coordinates of each blob center of gravity) and then, knowing the 3D coordinates of the points, estimate the pose of the object.

Other works that deal with pose estimation exploits some markers, such as the AprilTag marker [14] and the related AprilTag detection software that computes the precise 3D position, orientation of the tags relative to the camera.

However these techniques reduce to particular cases, in general on a plain garment we do not have neither markers nor coplanar dots nor trackers.

To maintain the generality of the solution, some projects (for example [15]) exploit the structure from motion technique. This method uses a series of 2-dimensional images of a scene to reconstruct the 3-dimensional structure of the present objects. The images can be taken from different cameras or performing small motions of the same camera.

2.4 Visual servoing

Visual servoing is a technique that uses information acquired by vision sensors to perform feedback control of the motion of a robot (or of some of its parts, like the end-effector).

There are several alternative schemes of visual servoing:

- position-based visual servoing;
- image-based visual servoing;
- 2-1/2-D visual servoing.

2.4.1 Position-based visual servoing

Position-based visual servoing, also known as 3-D visual servoing, is characterized by an input computed in the 3-D Cartesian space.
The feedback is based on the real-time estimation of the pose of the observed object with respect to the camera, this pose is estimated from image features corresponding to the perspective projection of the target in the image.

The estimation of the pose can be performed analytically or using iterative numerical algorithms, but all methods are based on the knowledge of a perfect geometric model of the object and necessitate a calibrated camera to obtain unbiased results.

The advantage of this scheme is that we can act directly on the operational space variables. The cons are that we always need a model of the object (that usually for cloths is not available) and that the object may exit from camera field-of-view, this would cause an open feedback loop and instability of the system.

2.4.2 Image-based visual servoing

The image-based visual servoing (also known as 2-D visual servoing) is characterized by an input computed in the 2-D image space. The control action is computed on the basis of the error defined as the difference between the value of the image feature parameters in the desired configuration and the value of the parameters measured with camera in the current pose.

The pros of this method are the fact that we do not need a real-time estimate of the pose of the object with respect to the camera and that the objects are always within the camera field-of-view. This scheme is robust against both camera and robot calibration errors. The cons are the fact that singular configurations may occur because of the non-linearity of the map between image feature parameters and the operational space variables; moreover the end-effector trajectory cannot be easily computed in advance, so there could be collisions or joint limits violations, and finally we cannot ensure the convergence of the control law. A good approach for this type of visual servoing scheme can be found in the project [**16**].

2.4.3 2-1/2-D visual servoing

At last, 2-1/2-D visual servoing is a new approach which combines the advantages of 2-D and 3-D visual servoing and avoids their drawbacks. It has been analyzed in the work of E. Malis, F. Chaumette, S. Boudet [**17**].

The input is expressed in part in the 3-D Cartesian space and in part in the 2-D image space, it is based on the estimation of the camera displacement (the rotation and the scaled translation of the camera) between the current and desired views of the object (at each iteration of the control law). The camera rotation between the two views is computed at each iteration, so the rotational control loop is immediately obtained.

To control the translational camera degrees of freedom some extended image coordinates of a reference point of the target are introduced, so we obtain a triangular interaction matrix. The Jacobian matrix has no singularity in the whole task space, so there is convergence of the positioning task for any initial camera position if the camera intrinsic parameters are known. Finally, if the camera intrinsic parameters are not perfectly known, the estimated control vector can be analytically computed as a function of camera calibration errors.

The main pros of this method are:

- we don't need a (3D) model of the object
- using an adaptive control law, we can ensure that the target will always remain in the camera field of view
- we can ensure convergence of the control law (because the Jacobian matrix has no singularities)
- the scheme is robust against camera calibration errors.

To track more accurately the pose of the manipulator, the project [**18**] exploits virtual visual servoing. This approach aligns a rendered model of the robot parts with the current image of their real counterparts.

The procedure starts performing a scene reconstruction using the head of the robot to explore the environment, then the scene is segmented into object hypotheses and background. After that a table plane is detected and the hypothetic objects are placed on it, obtaining a scene model. Finally a suitable grasp is selected for each object hypothesis depending on the available knowledge about the object.

Another approach consist in formulating the visual servoing in a quadratic optimization framework [**19**].

Indeed visual servoing techniques try to solve the system of equations:

$$\begin{cases} \dot{e} = L_e v & \text{(1.a)} \\ v = -\lambda \widehat{L_e^+} e & \text{(1.b)} \end{cases}$$

where e is an error vector of the visual features, v is the velocity of the camera, λ is a gain, L_e is the interaction matrix and $\widehat{L_e^+}$ the pseudoinverse of its estimate.

The idea of this work is that it is necessary to use Equation (1.a), while Equation (1.b) can be rewritten as an optimization problem, for example:

$$v = argmin||v||^2 \tag{2.a}$$

$$\text{subject to} \quad \widehat{L_e} v = -\lambda e \tag{2.b}$$

This allows to face visual constraints (like occlusion avoidance), as inequalities.

ETSEIB

2.5 Cloth region segmentation

Cloth region segmentation is a procedure that processes the area of the image corresponding to the garment, extracting some interesting features. Nowadays, vision methods are mainly based on deep learning techniques. The community is rapidly proposing new methods, and it is foreseen that powerful segmentation methods will shortly become available.

In [**20**], a convolutional network based on the popular Hourglass architecture is used to detect the corners of a garment lying on a table. The method is very sensitive to the point of view of the camera due to the small training set used.

In a more generic approach [**21**], a neural network is trained to predict where not only the corners but also the edges of a cloth are located distinguishing these regions from wrinkles or folds. The network predicts where the corners, the outer and inner edges of the cloth are located from a depth image of the scene. To train such a network, ground-truth labels for the cloth edges and corners are needed. To avoid a large amount of human annotation effort, all the edges and corners have been marked with different colours so that the problem could be treated as a semantic segmentation.

Based on this segmentation, in the same paper it is shown how to compute an estimate of the grasp direction and of the direction uncertainty, used to prevent from fails due to a wrong approaching direction. After that the grasp point is selected among all the points as the one belonging to the outer edges which has the lowest direction uncertainty.

Once a the grasp points and direction are chosen, the approach to the cloth is performed sliding one of the gripper's fingertips under the cloth for a pinch grasp. Figure 2 shows an example of cloth region segmentation (a), a cropped section of the previous image (b), a subset of grasp direction proposals for each point belonging to the outer edges (c) and finally the grasp direction uncertainty for each of the points belonging to the outer edges (d).

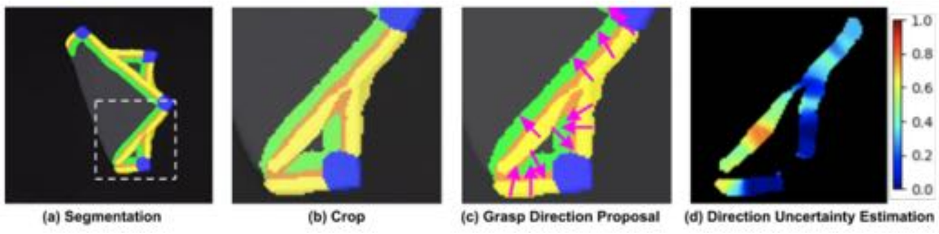
1.0
0.8
0.6
0.4
0.2
0.0
(a) Segmentation
(b) Crop
(c) Grasp Direction Proposal
(d) Direction Uncertainty Estimation

Figure 2: Illustration of grasp configuration selection, image extracted from [**21**]

2.6 Grasping

The main difference between grasping a rigid and a deformable object is that in the first case is required a soft hand (like the ones shown in Figure 3), able to adapt to the shape of the object, while in the second case is the garment that adapts to the shape of the end-effector (some examples are reported in Figure 4). Moreover in the case of deformable objects extrinsic contacts are exploited to improve the grasp success, while during manipulations of rigid objects usually they are avoided.

A recent review that deals with grasps of flexible objects can be found in [**5**]. This work classifies the types of grippers and grasps already present in literature to perform cloth manipulation, and analyzes the manipulation primitives based on regrasps. Finally an experiment is developed to calculate the influence of the grasp type on task performance.

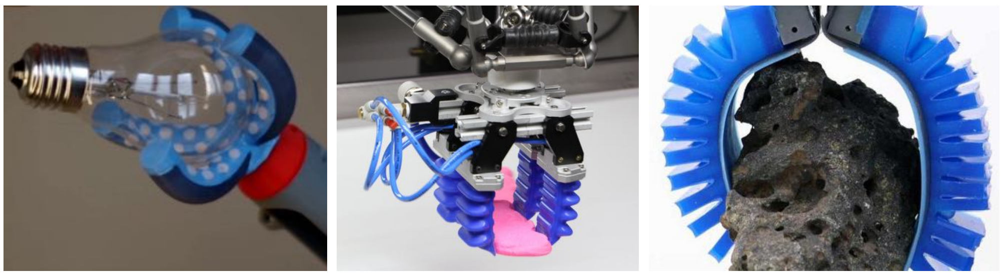

Figure 3: Examples of soft grippers. a) gripper designed at Industrial Research Institute for Automation and Measurements in Warsaw [**22**], b) gripper designed by Soft Robotics and inspired to octopus [**23**], c) gripper with Gekco-inspired adhesives [**24**].

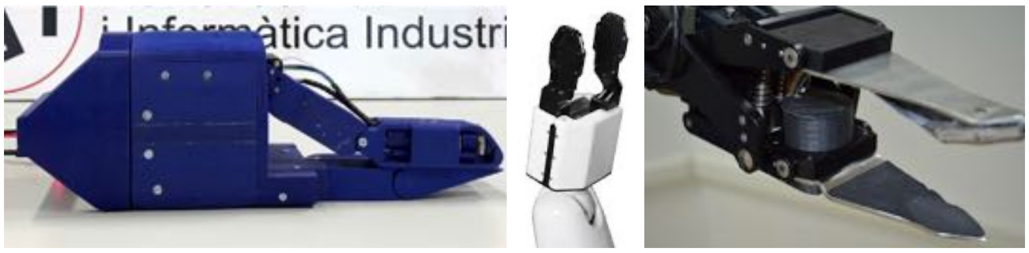

Figure 4: Examples of rigid grippers. a) gripper designed at Institut de Robòtica i Informàtica Industrial in Barcelona [**25**], b) gripper designed by Pal Robotics for TIAGo robot [**26**], c) gripper designed by Information Technology Institute in Thessaloniki [**27**].

3 Robotic system description

3.1 Hardware and software characteristics of the system

For the experiments related to this project has been used a TIAGo robot at the Institute of Robotics and Industrial Informatics (IRI) in Barcelona.

TIAGo (Take It And Go) is a service robot designed by the company PAL Robotics to work in indoor environments, the tests have been performed in a laboratory simulating an apartment.

The robot, shown in Figure 5, presents many capabilities such as mobility, perception, manipulation and human-robot interaction capabilities, in the following we provide a description of its architecture from the hardware and the software point of view.

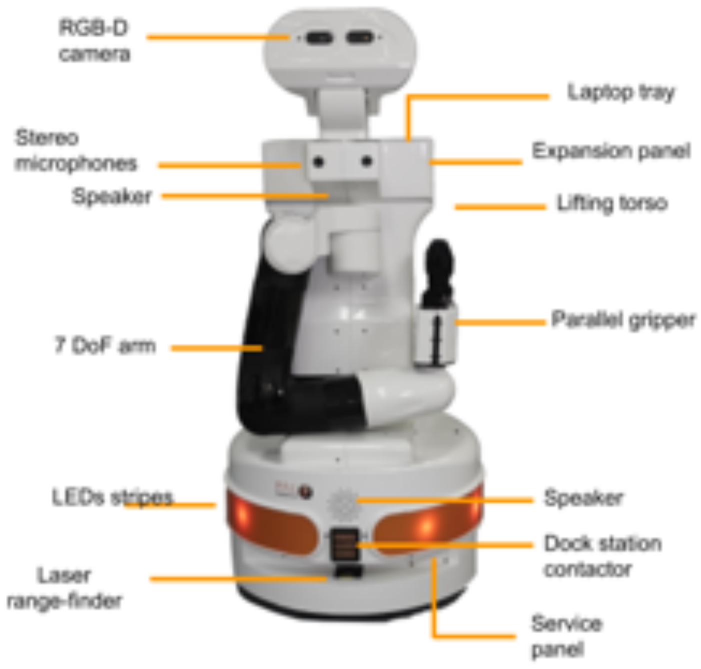

Figure 5: TIAGo's main components

From the hardware point of view TIAGo can be divided in five parts: the mobile base, the torso, the arm, the end-effector and the head.

Starting from the bottom, the robot has a mobile base, with a differential drive, that contains all the necessary elements for navigation and obstacle avoidance.

On the top of the mobile base there is a prismatic joint that allows to lift or lower the torso changing the height of the robot from a minimum of 110 cm to a maximum of 145 cm. The torso supports the robot's arm and head and it includes an expansion panel and a laptop tray.

The arm provides a total of 7 degrees of freedom, it is composed by 4 modules and a wrist (that provides 3 degrees of freedom) with a force/torque sensor integrated. On the wrist two types of end-effectors can be mounted: a parallel gripper or a five-finger hand. For this work has been used a parallel gripper designed at the Institut de Robòtica i Informàtica Industrial (IRI). This end-effector, shown in Figure 6, has a flat structure with a very thin tip that makes it able to insert between the layers of a folded garment.

On this end-effector has been embedded an endoscopic RGB camera that was used to perform the movement from the pre-grasp position to the grasp position with more precision.

TIAGo's head has 2 degrees of freedom because it can be moved using two motors that provide a pan-tilt mechanism. These pan-tilt movements are important because they increase the visual range of the perception area without moving the robot. On the head there are stereo microphones, a speaker and an RGB-D camera.

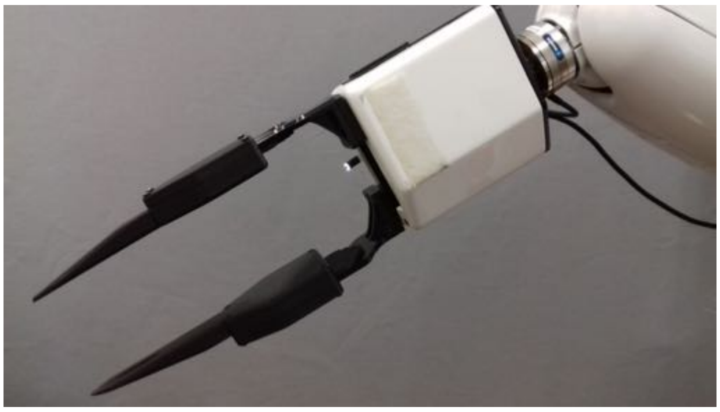
Figure 6: End-effector used for this project

From the software point of view, for this project has been used the 64-bits version of Ubuntu operating system. Then the most important software resource used are Robotic Operating System (ROS) and C++.

3.1.1 Robot Operating System

Robot Operating System (ROS) is a communication interface that has been used as a middleware between the software layer and the operative system. It provides the services of an operating system (including hardware abstraction, low-level device control, implementation of commonly-used functionality), the capability to communicate with an anonymous system of publishing and subscribing messages and many other tools and libraries.

ROS relies on the concepts of nodes and messages. A node or package is an executable that communicates with other nodes through some messages, that are ROS data types used to publish or subscribe a topic. There are three types of messages:

- Topics, that are used for a bidirectional communication between a Publisher node, that transmits information, and a Subscriber that receives the message. This type of communication is asynchronous and the Publisher does not know if data have been correctly received or not.
- Services, that rely on couples of messages: one message is for making a request and the other is for the reply in a server/client communication.
- Actions, that use ROS topics to send messages from a client to the server.

Thanks to this system data can be passed between processes and extracted without changes to code.

3.1.2 IRI's modular structure

The use of ROS services and actions implies the presence of a huge amount of variables, callbacks and functions; moreover there are two state machines, one for the client and one for the server, for each action and service.

In order to simplify this scenario, a structure based on modules has been implemented at IRI. The modules implement one state machine each (related to a robot functionality) that manages the interfaces of the actions and services involved, so the modules can be used as libraries in other applications reducing the number of messages used directly.

In particular some of the implemented modules that could be exploit to perform the movement from the starting position to a pre-grasp position, are the ones related to the arm, the head, the torso and the motion of the platform.

3.1.3 Whole body controller

In order to perform the motion from the pre-grasp position to the grasp position we exploit the Whole Body Control implemented by PAL Robotics. The Whole Body Control (WBC) is a quadratic hierarchical solver that provides the inverse kinematics of the whole robot body. It is able to accomplish different tasks with priorities, the tasks with highest priority are joint limit avoidance and self -collision avoidance.

This means that the user can command a task, for example moving the arm-tool link to a desired spatial configuration, but this task has a lower priority, so the solution will always avoid to reach joint limits and will prevent the arm from colliding with any other part of the robot while moving.

The difference between the Whole Body Controller and standard inverse kinematic solvers is that the former finds automatically solutions that guarantee self-collisions avoidance and joint limit avoidance.

To complete the tasks the WBC takes control of TIAGo's torso, arm and head. This can be noticed in Figure 7 where we can see that in order to reach the position $(0.5, 0.0, 0.9)$ and orientation $(0°, 30°, 0°)$, the WBC moves not only the arm, but also the torso; in this case the head does not move.

3.2 Perception of garment

The first phase of the process is the detection of the garment. This phase has been developed in previous projects as follows.

In the project [**12**], the garment is detected with a technique based on segmentation. This method locates the grasping point with a not very high accuracy, but sufficient for the pre-grasp motion. First of all, with the information provided by TIAGo's RGB-D camera, a segmentation is performed to separate the biggest horizontal plane from the objects located on it. The segmentation assigns the points of the image to two different clusters according to a certain threshold.

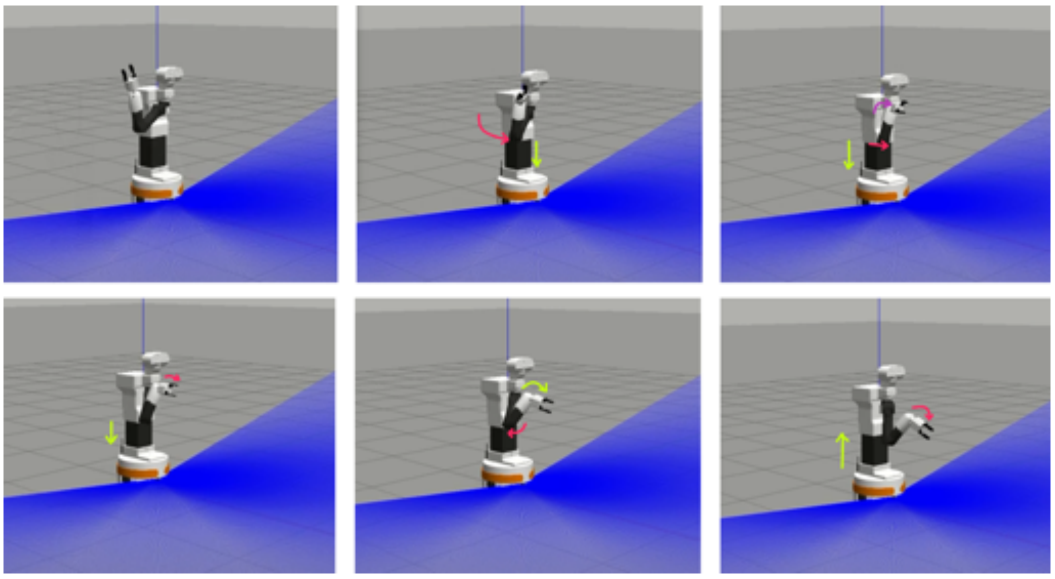
Figure 7: WBC controls TIAGo's torso and arm to reach the configuration

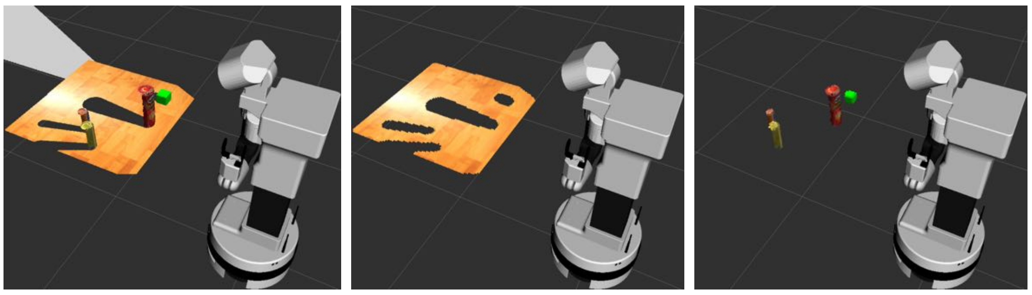
a) b) c)

Figure 8: Examples of segmentation, a) Scene perceived by TIAGo, b) table point cloud, c) objects point cloud

The result of this process is that the input point cloud is split in two clusters, one that contains the points that belong to the table and the other with the points of the cloth placed on it. Figure 8 shows an example of the segmentation results with TIAGo placed in front of a table with several objects.

Henceforth the cluster of the table points is considered only as an obstacle to avoid collisions and the focus is on the cluster of points of the garment.

3.3 Motion pre-grasp

Once the garment has been detected, TIAGo's arm can be moved towards the garment. As shown in Figure 9, the whole movement of the arm from the initial position to the end, can be divided into three parts:

- movement from the initial position to a position called of "pre-grasp",
- movement from the pre-grasp position to the grasping point,
- movement from the grasping point to a position called of "post-grasp".

This section describes how can be performed the first part of the movement, that brings the end-effector to a position close to the corner. This phase improves the precision of the grasping phase, because then the movement from the pre-grasp position to the grasping point is very short and more linear.

To approach the garment it is only needed to avoid collisions with the table, so it is not required a high precision. On the contrary, the following movement from the pre-grasp position to the grasping point has to be very accurate, in order to perform fine grasping. The final motion towards the post-grasp position has the only objective to separate the end-effector and the cloth from the table, so it does not require to be particularly accurate.

The pre-grasp movement consists in planning and performing a joint trajectory to bring the end-effector to a chosen position and orientation in space, the pre-grasp position.

However, as we can see in Figure 10, the *gripper_grasping_frame*, that is the reference frame corresponding to the end-effector, does not coincide with the *arm_tool_link* reference frame, placed on the wrist of the arm, that is the frame that we can actually move exploiting TIAGo's WBC or modules.

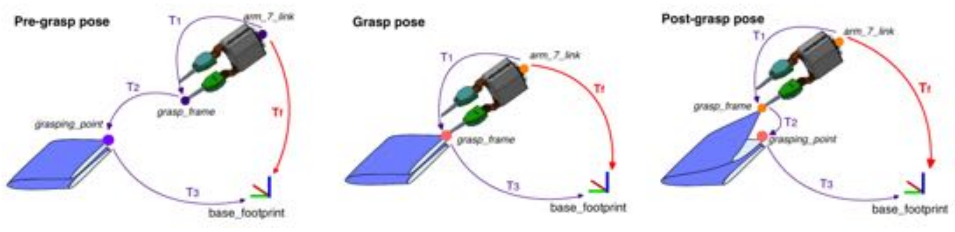

Figure 9: Steps of the grasping process, image extracted from [**12**]

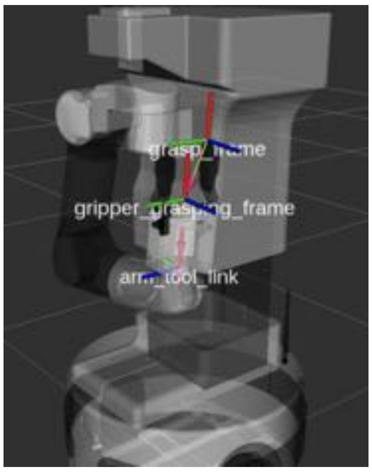

Figure 10: Reference frames on TIAGo's arm, image extracted from [**12**]

So we have to plan a joint trajectory to bring the *arm_tool_link* reference frame to a position such that the gripper is in the desired pre-grasp position.

First we create a reference frame, called *grasp_frame*, that corresponds to the grasp point of the gripper. The transformation T_f between the position of the *arm_tool_link* with respect to TIAGo's base frame, can be computed as:

$$T_f = T_3 \cdot T_2 \cdot T_1.$$

where:

- T_1, is the transformation between the *arm_tool_link* frame and the *grasp_frame*;
- T_2, represents the position of the *grasp_frame* with respect to the corner position;
- T_3, represents the corner position with respect to the *base_footprint* reference frame that is the frame of the mobile platform.

Then this transformation can be exploited to bring the *grasp_frame* to the pre-grasp position exploiting the modules.

4 Development of visually guided grasping

Once reached the pre-grasp position, the goal is to move the gripper in order to insert one of its fingertip under the first layer of the cloth. From previous works and literature we know that image analysis from a camera in the head of the robot has not the proper view-point and thus, the information provided to the robot control cannot be precise enough. Therefore, to improve the precision of this movement it has been decided to use an endoscopic camera mounted on the end-effector of the robot.

4.1 Evaluation of some visual approaches

Many different ways to reach the corner, exploiting the information given by the endoscopic camera, have been analyzed, first of all the visual servoing, then structure from motion and object tracking and finally corner detection.

4.1.1 Visual servoing

This technique requires the knowledge of the 3D coordinates of four points belonging to the object, necessary to compute the visual features, and the goal pose, that in our case coincides with the position and orientation of the corner.

One way to recover the 3D pose of the corner consists in drawing 4 coplanar points on the garment and exploiting the knowledge of the relative positions of these points to compute the pose.

Indeed the homogeneous transformation matrix corresponding to the relative pose of the garment with respect to the camera, can be defined as:

$$\mathbf{T_o^c} = \begin{pmatrix} \mathbf{R_o^c} & \mathbf{o_{c,o}^c} \\ \mathbf{0^T} & 1 \end{pmatrix}$$

where:

$$\mathbf{o_{c,o}^c} = \mathbf{o_o^c} - \mathbf{o_c^c}$$

and we define:

- $\mathbf{o_c^c}$ the position vector of the origin of the camera frame with respect to the base frame, expressed in camera frame,

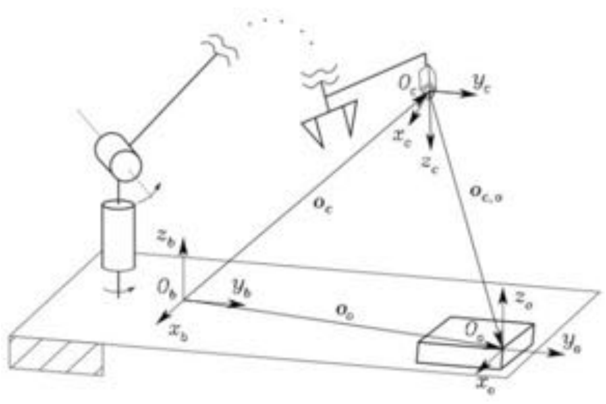

Figure 11: Frames and position vectors

- $\mathbf{o_o^c}$ the position vector of the origin of the object frame with respect to the base frame, expressed in the camera frame,
- $\mathbf{R_o^c}$ the rotation matrix of the object frame with respect to the camera frame.

Figure 11 represents these frames and position vectors. The components of matrix R_o^c are:

$$r_1 = \xi h_1$$
$$r_2 = \xi h_2$$
$$r_3 = r_1 \times r_2$$

and we have:

$$\mathbf{o_{c,o}^o} = \xi h_3$$

where h_i denotes the i-th column of the matrix $\mathbf{H}$.

Defined $\mathbf{r_{o,i}^o}$ $=\mathbf{r_i^o} - \mathbf{o_o^o}$, i=1,...,4 the position vectors of the points with respect to the object frame, and $r_{x,i}$ and $r_{y,i}$ their two non-null components, matrix $\mathbf{H}$ derives from:

$$S(\widetilde{s}_i)\mathbf{H}\begin{pmatrix} r_{x,i} & r_{y,i} & 1 \end{pmatrix}^T = 0.$$

The previous expression is obtained multiplying by the skew-symmetric matrix $\mathbf{S}$ both sides of equation:

$$\lambda_i \widetilde{s}_i = \prod \mathbf{T_o^c} \widetilde{r^o}_{o,i}$$

that represents the homogeneous coordinates of the projections on the image plane of the points of the object with respect to the camera frame.

However, this procedure works correctly when dealing with rigid objects, but is not applicable in the setup of our experiment. Indeed it is difficult to univocally locate the four points in space and ensure their coplanarity considering the deformable nature of the garments. It was therefore decided to develop a different strategy.

4.1.2 Structure from motion and object tracking

Another possibility is to recover the 3D position and orientation of the corner and then reach this pose performing object tracking. A way to recover the 3D position of the corner without imposing particular conditions (such as the presence of 4 known coplanar points) is to exploit structure from motion.

Structure from motion is a technique that, given a sequence of several images taken moving the camera around an object, tries to recover the 3D structure of that object. To do so, it extracts keypoints from each image and finds the matches between subsequent images, recovering the depth of the common points; in this way it recovers some 3D points of the object.

A script has been implemented to apply this technique and many tests have been performed, with the available endoscopic camera, to verify the validity of this approach. In particular the tests have been made putting the garment on a light-gray table and varying the following variables:

- the distance between the endoscopic camera and the corner of the garment, that has been set equal to 5 cm, 10 cm or 20 cm;
- the angle between the endoscopic camera and the table, that has been set equal to 0° or 45°;
- the angles on the xy-plane (coincident with the table) representing the rotations of the camera images, these angles have been set equal to 0°/15°/30°/45° or 0°/30°/60°/90°.

The scheme reported in Figure 12 summarizes the tests performed with the white garment and Figure 13 shows the setup of the tests with the white cloth.

These tests have been performed also with a textured garment, to see if the texture would imply better results, thanks to the higher number of keypoints. The scheme reported in Figure 14 summarizes the tests performed and Figure 15 shows the setup of the tests with the textured garment.

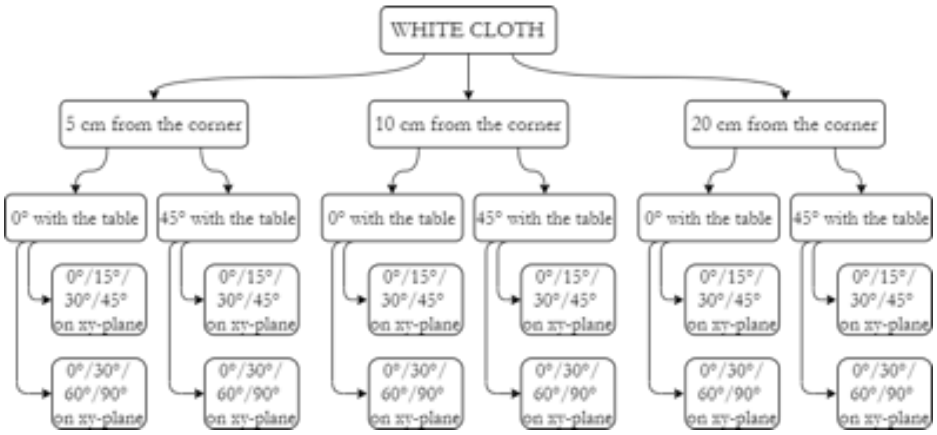

Figure 12: Scheme of the tests performed with the white cloth

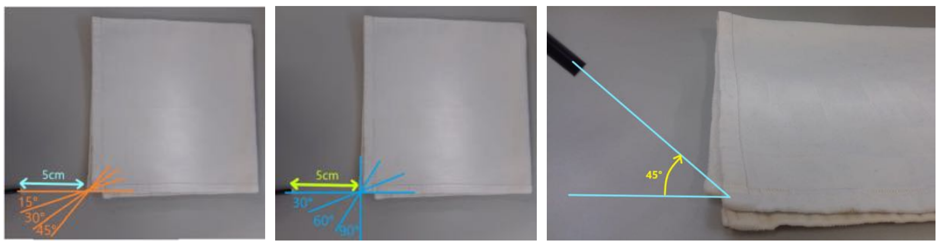

Figure 13: Setup of the tests with the white cloth

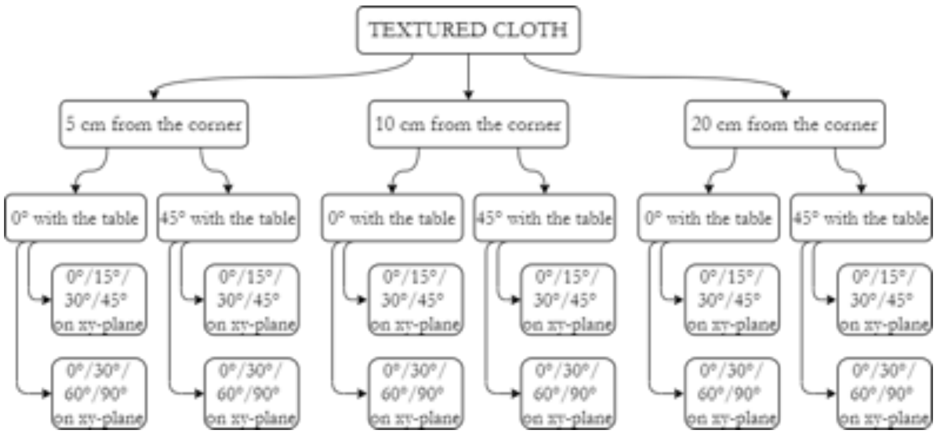

Figure 14: Scheme of the tests performed with the textured cloth

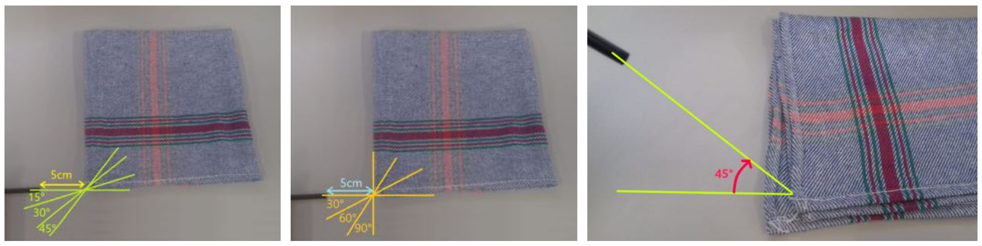

Figure 15: Setup of the tests with the textured cloth

What we found out from these tests is that the resolution of the images is too low to be able to find matches between subsequent images and consequently to reconstruct the positions of the points in space.

This is due to the fact that the endoscopic camera is designed to work better in low light conditions and has a very short focal length. Moreover usually garments have a low texture which makes difficult to detect enough keypoints to apply this technique.

In many cases no 3D points were reconstructed, while in the cases when some points have been reconstructed (as the ones reported in Figures 16, 17, 18 and 19), the number of points was not sufficiently high to provide a consistent 3D reconstruction of the object.

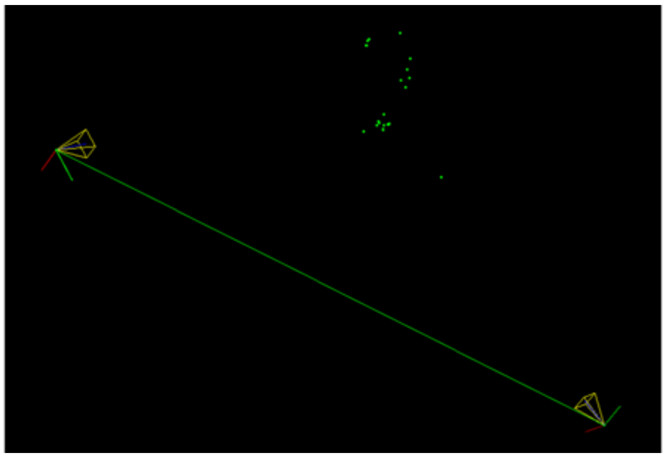

Figure 16: Test with the textured cloth, images were taken at 10 cm from the corner, angle of 0° with the table and angles of 0°, 30°, 60°, 90° on xy-plane

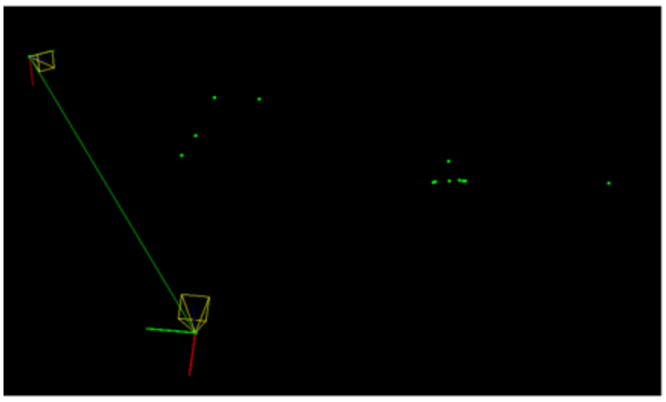

Figure 17: Test with the textured cloth, images were taken at 10 cm from the corner, angle of 0° with the table and angles of 0°, 30°, 60°, 90° on xy-plane

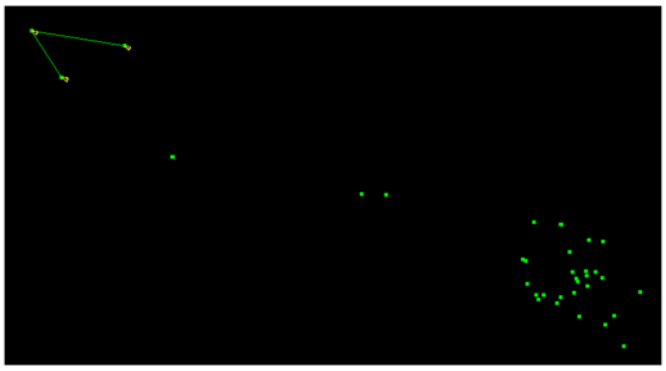

Figure 18: Test with the textured cloth, images were taken at 10 cm from the corner, angle of 0° with the table and angles of 0°, 15°, 30°, 45° on xy-plane

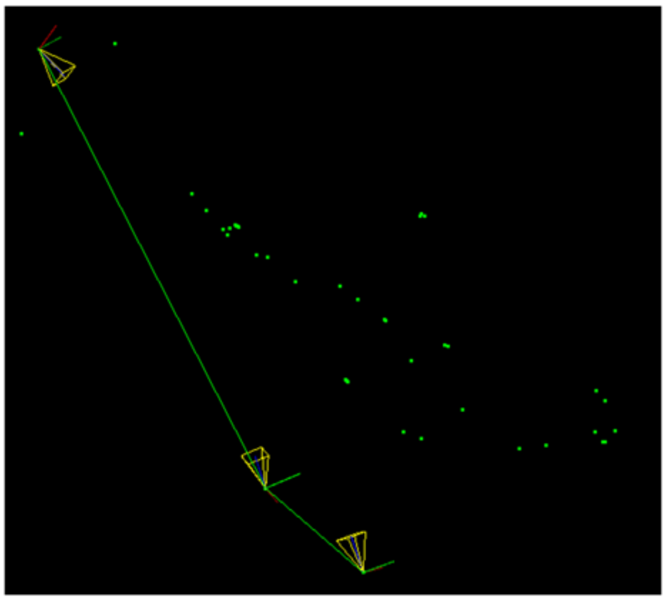

Figure 19: Test with the textured cloth, images were taken at 10 cm from the corner, angle of 0° with the table and angles of 0°, 15°, 30°, 45° on xy-plane

In the tests previously shown the number of estimated points was 22, 12, 37 and 46 respectively, we can also notice that the estimated number of positions of the camera was always 2 or 3 instead of 4.

For each case more tests have been performed, the results found have been reported in Table 1.

Type of garment	**Distance from corner**	**Angle with the table**	**Angles on xy-plane**	**Number of tests**	**Max number of 3D points**
White	5	0°	0°/15°/30°/45°	5	11
White	5	0°	0°/30°/60°/90°	4	0
White	5	45°	0°/15°/30°/45°	3	0
White	5	45°	0°/30°/60°/90°	3	0
White	10	0°	0°/15°/30°/45°	4	0
White	10	0°	0°/30°/60°/90°	4	0
White	10	45°	0°/15°/30°/45°	3	0
White	10	45°	0°/30°/60°/90°	3	0
White	20	0°	0°/15°/30°/45°	3	0
White	20	0°	0°/30°/60°/90°	3	0
White	20	45°	0°/15°/30°/45°	3	0
White	20	45°	0°/30°/60°/90°	3	0
Textured	5	0°	0°/15°/30°/45°	4	12
Textured	5	0°	0°/30°/60°/90°	4	0
Textured	5	45°	0°/15°/30°/45°	3	9
Textured	5	45°	0°/30°/60°/90°	3	0
Textured	10	0°	0°/15°/30°/45°	4	46
Textured	10	0°	0°/30°/60°/90°	4	22
Textured	10	45°	0°/15°/30°/45°	3	0
Textured	10	45°	0°/30°/60°/90°	3	0
Textured	20	0°	0°/15°/30°/45°	3	0
Textured	20	0°	0°/30°/60°/90°	3	0
Textured	20	45°	0°/15°/30°/45°	3	0
Textured	20	45°	0°/30°/60°/90°	3	0

Table 1: Results of structure from motion tests

Since the endoscopic camera has been designed to work better in low light conditions, we tried to perform some tests putting the garment on a dark surface, to see if the loss of sharpness in the images could be caused also by reflections due to the light color of the table and the lights of the room.

What we saw is that the result improved, but it still was not sufficiently robust; moreover we would like to find a solution as general as possible, without making assumptions on the appearance of the environment (like the colour of the table).

Therefore we can conclude that this technique is valid from the theoretical point of view, but it is not applicable with the available hardware resources.

We observe that the resolution of the images generated by the endoscopic camera is sufficient to perform corners and edge detection, thus we have experimented two additional procedures presented in next sections.

4.1.3 Approach based on corner detection and tracking

Given the images coming from the endoscopic camera, corners can be detected exploiting Harris or Shi-Tomasi corner detectors. Once detected the corners we identify the one relative to the upper level of the garment and we track its position during the motion.

The Harris corner detector interprets a corner as the junction of two edges, where an edge is a sudden change in image brightness. The idea is to consider a small window around every pixel in the image and compute a score for each pixel:

$$score = \det(\mathbf{M}) - k \cdot (trace(\mathbf{M}))^2$$

where $\mathbf{M}$ is the auto-correlation matrix of the window, $\det(\mathbf{M}) = \lambda_1 \cdot \lambda_2$ and $trace(\mathbf{M}) = \lambda_1 + \lambda_2$, where λ_1 and λ_2 are the eigenvalues of the matrix $\mathbf{M}$.

Then we have that:

- when $|score|$ is small, so λ_1 and λ_2 are small, the region is flat,
- when $score < 0$, which means that $\lambda_1 >> \lambda_2$ or vice versa, the region is an edge,
- when $score$ is large, so λ_1 and λ_2 are large and $\lambda_1 \sim \lambda_2$, the region is a corner.

Shi-Tomasi corner detector is based on Harris corner detector, but has a variation in the selection criteria that makes it succeed where Harris detector fails. The difference is that Shi-Tomasi corner detector computes the score as:

$$score = \min(\lambda_1, \lambda_2).$$

In this way a pixel can be marked as a corner if the score is greater than a certain threshold.

In the following are reported some tests made exploiting Shi-Tomasi corner detector. As can be noticed from Figures 20 and 21, this approach is not reliable.

Indeed its performance depends on the light intensity, on the perspective and on the way the cloth is folded, therefore it is not possible to find values for the parameters that guarantee to extract only the position of the corner of interest maintaining the generality.

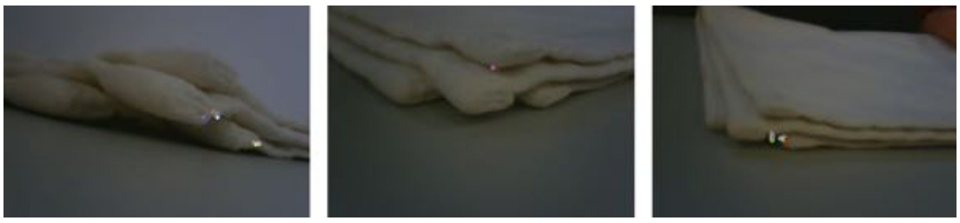

Figure 20: Examples of corner detection with images taken from different perspectives

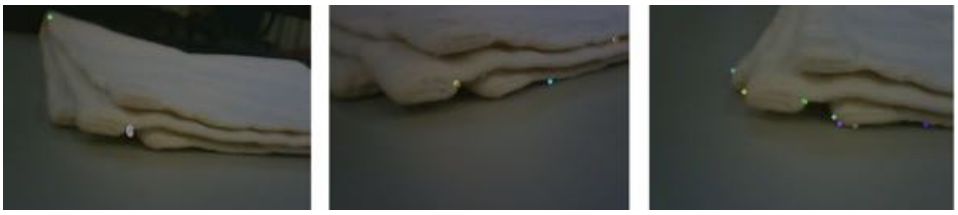

Figure 21: Examples of corner detection with images taken from different perspectives

Since even this approach is not enough robust, we decided to look no more for the corners, but for the lines in the image and we proceeded as follows.

4.2 Visual approach developed in this project

We integrate the endoscopic camera into the end-effector of the robot, so that in the images we see the final part of the gripper, which appears as a rectangle fixed at the bottom of the image, and the garment to be grasped.

Then from the images we extract the edges and we detect the lines, in particular we are interested in the horizontal lines which correspond to the various layers of the garment.

Once detected the lines we identify the one relative to the lower side of the upper level of the garment (see Figure 22 for more clarity).

Our goal is to insert the lower finger of the gripper between the upper and the second layer of the garment, so we implement a controller to move the robotic arm towards the garment according to these rules:

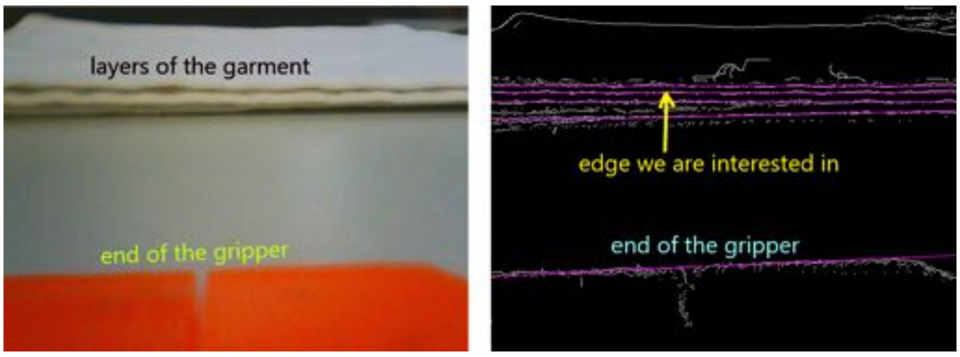

Figure 22: Example of image once the camera is integrated in the gripper

- in the image the edge relative to the final part of the gripper must remain exactly below the edge corresponding to the upper layer of the cloth;
- the arm-tool link has to move to the position:

$$(x = old\_x + \delta x,\ y = old\_y,\ z = old\_z + \delta z),$$

  where the increments δx and δz depend on the relative position of the end-effector with respect to the edge of the upper layer of the garment;
- roll, pitch and yaw angles are fixed;
- the movement continues until the edge of the end-effector is no longer visible in the image, which means that the garment has been reached (and it hit the camera).

This means that we have to implement a controller that keeps the gripper under the edge relative to the upper layer of the garment, so its position is wrong both if it is too high (so the edge is no more visible over the gripper) and if it is too low (so too many edges are visible over the gripper).

4.2.1 Segmentation of the image

The image has first been segmented using Canny edge detector. This technique extracts the edges of the image performing 5 steps:

- Gaussian filtering to smooth the image and remove some noise;
- computation of the magnitude and phase of the gradients;
- quantization of the gradients angles;

- non-maxima suppression to thin the edges;
- hysteresis thresholding, that consists in double thresholding to suppress all the weak edges not connected to strong edges.

Figures 23, 24, 25 and 26 show the results of segmentation applied to images of a white and a textured cloth taken from different distances and perspectives.

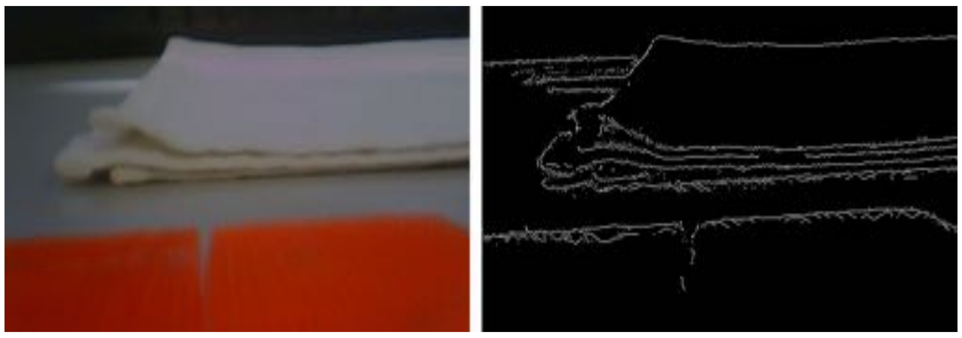

Figure 23: Example of Canny segmentation on the image of a white cloth

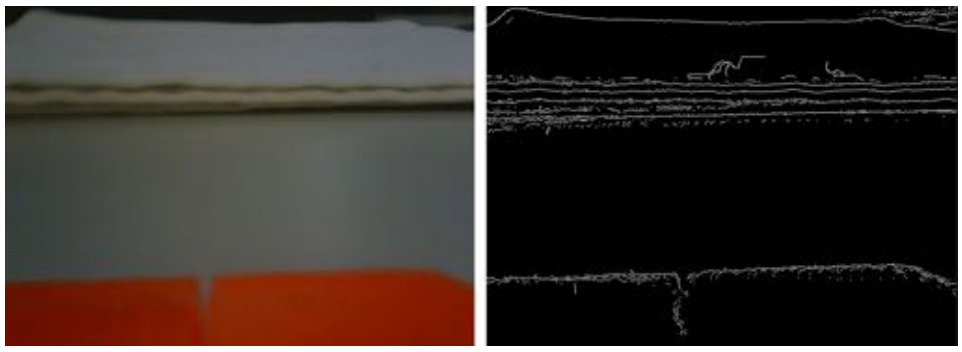

Figure 24: Example of Canny segmentation on the image of a white cloth

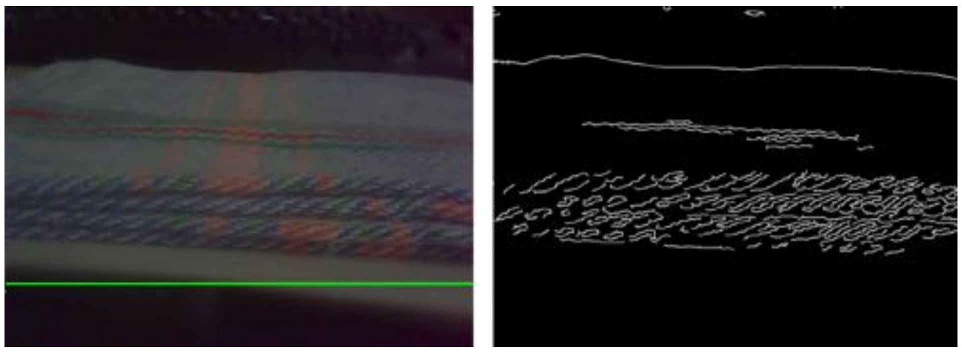

Figure 25: Example of Canny segmentation on the image of a textured cloth

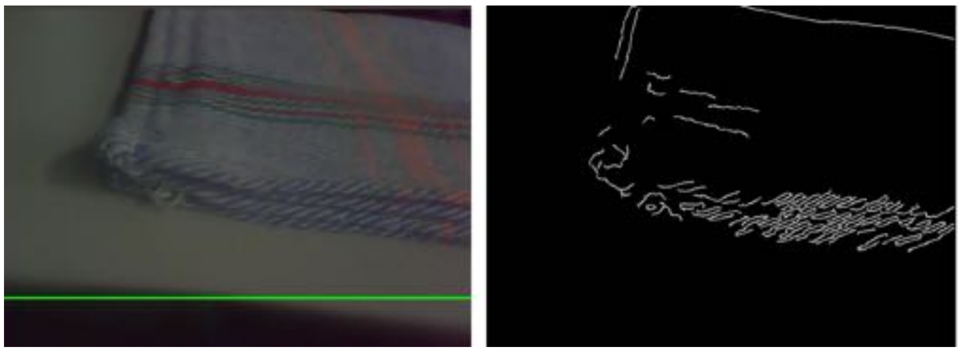

Figure 26: Example of Canny segmentation on the image of a textured cloth

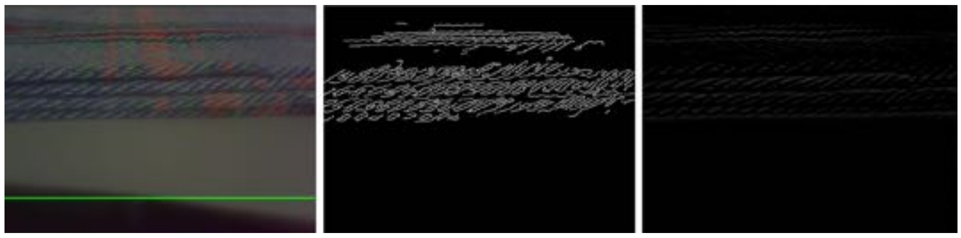

Figure 27: Original image of a textured cloth, effect of Canny edge detector, effect of Canny edge detector and Sobel filter

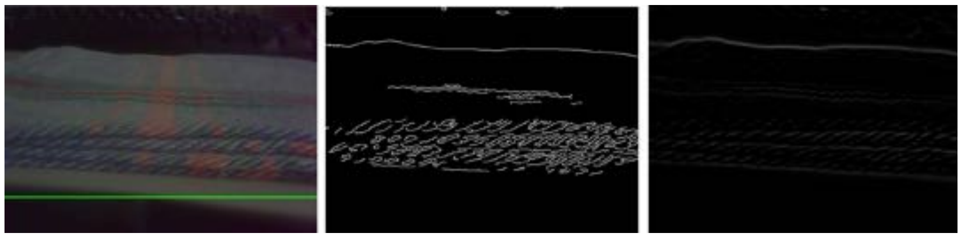

Figure 28: Original image of a textured cloth, effect of Canny edge detector, effect of Canny edge detector and Sobel filter

The results of segmentation are informative enough, but, in particular in the case of the textured cloth, there is still a lot of noise due to non-horizontal edges. Therefore a Sobel filter has been applied to filter out the non-horizontal edges. This operator convolves a kernel with the original image in order to calculate an approximation of the derivative for horizontal or vertical changes. The kernel size can be chosen equal to 3, 5 or 7, in our case it has been set equal to 3 because with higher sizes the edges are too thick and there is too much noise. If we want to preserve horizontal edges and discard all the others, we have to take a kernel that contains the vertical derivative approximation,

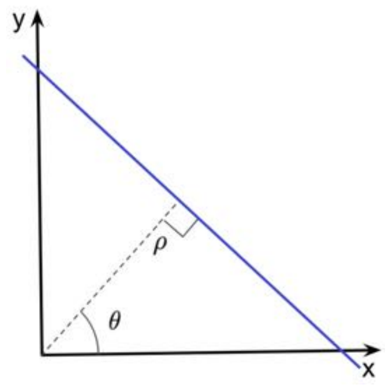

Figure 29: Normal representation of a straight line

which means to take:

$$\mathbf{G_y} = \begin{pmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

We always choose the derivative orders equal to 0 in x direction and 1 in y direction, because we want to highlight only horizontal edges and higher order derivatives are too sensitive to noise.

Figures 27 and 28 show the aspect of the image after having applied only Canny edge detector (central images) and both Canny edge detector and Sobel filter (right images).

Comparing figures 23-26 with figures 27 and 28 we can notice that after Sobel filtering there is a lot less noise and the texture of the garment is not detected.

4.2.2 Line extraction

After the segmentation we have to detect the lines in the image. To do so we exploit Hough transform, a feature extraction technique that expresses each straight line in the normal form:

$$\rho = xcos(\theta) + ysin(\theta)$$

where ρ is the distance from the origin to the closest point on the line, and θ is the angle between the x axis and the line connecting the origin with that closest point (see Figure 29).

ETSEIB

Therefore Hough transform associates each line of the image with a pair (ρ, θ) through a voting procedure that consists in:

- creating a 2D array called accumulator,
- for every pixel (x, y) in the image, determining if there is enough evidence of a straight line at that pixel,
- if so, computing the parameters (ρ, θ) of that line,
- incrementing the value of the bin of the accumulator where the couple of parameters falls into,
- looping through all the values of the bins in the accumulator, if a value is larger than a certain threshold, then the corresponding couple (ρ, θ) identifies a straight line.

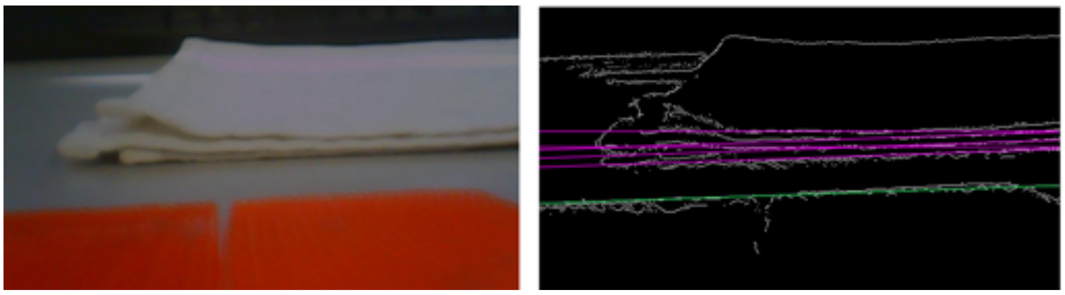

Figure 30: Hough lines on images of a white cloth where has been applied only the Canny edge detector

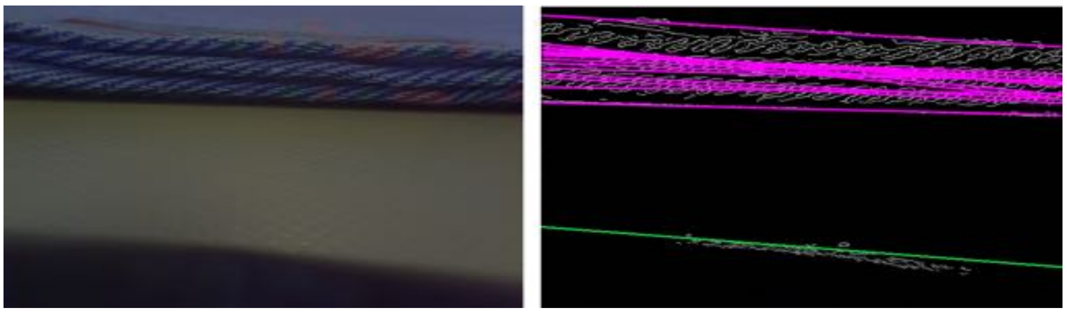

Figure 31: Hough lines on images of a textured cloth where has been applied only the Canny edge detector

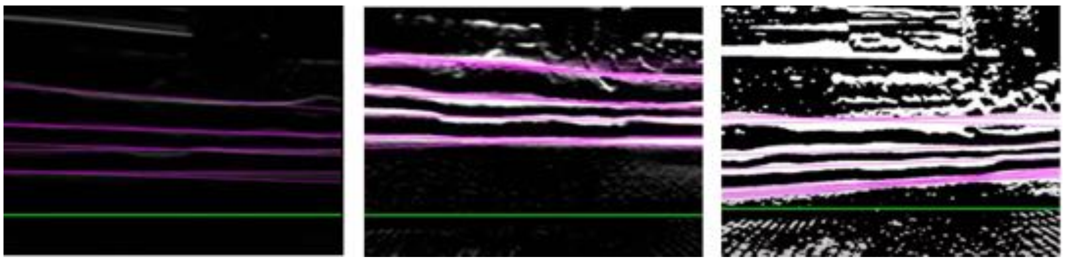

Figure 32: Hough lines on images of a white cloth where have been applied first the Canny edge detector and then a Sobel filter with kernel size equal to 3, 5 and 7 respectively

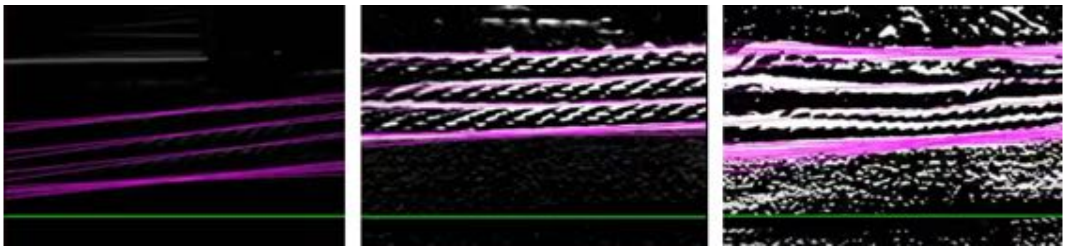

Figure 33: Hough lines on images of a textured cloth where have been applied first the Canny edge detector and then a Sobel filter with kernel size equal to 3, 5 and 7 respectively

To have a visual feedback, the lines have then been drawn in pink on the segmented image, while the border of the end-effector has been highlighted with a green line.

Figures 30 and 31 show the result of Hough line detection without applying Sobel filter on the images of a white and of a textured cloth respectively. Figures 32 and 33 show the result of Hough line detection applying a Sobel filter with different kernel sizes on the images of a white and of a textured cloth respectively.

From these figures we can notice that when we apply a Sobel filter (it is more evident with a smaller kernel size), the texture of the garment does not affect the performance of the edge detector.

4.2.3 Detection of the edge of interest

Now we want to select the line of interest, that is the one corresponding to the lower edge of the upper layer of the folded garment. To do so, we assume that

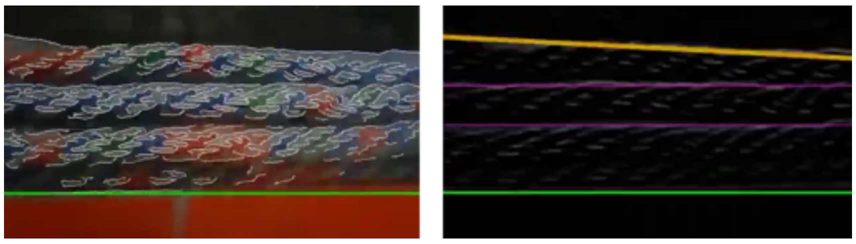

Figure 34: First two strongest lines of an image of a textured cloth

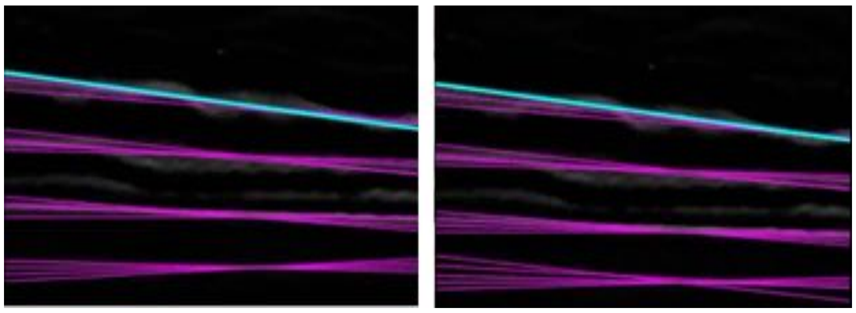

Figure 35: Phase of line detection in the first 5 frames

at the pre-grasp position the gripper has a pitch angle of approximately 30° in order to include mainly the table and the garment in the image, because we want to avoid having horizontal lines that belong to the background.

So we are interested in the second level of lines in the image, but Hough transform returns a family of almost coincident lines for each level. The idea to overcome this problem is to exploit the fact that Hough transform returns the detected lines in a vector called *lines* in order of strength. The strongest line corresponds to the border of the gripper, while the second and the third strongest lines usually correspond to the edges between the lower layer of the garment and the table and to the upper layer of the garment.

Figure 34 is an example where the strongest line has been drawn in green and the second strongest in yellow. So the lines that correspond to the intermediate layers are stored at higher indices of the vector *lines*. Since we cannot determine a priori to which layer the line of interest corresponds to, we select it choosing the line with the second highest y coordinate among the 20 strongest lines (the lines are almost horizontal, so we take the y coordinate that corresponds to x=0). To remove some noise in Canny image the Gaussian blurring has been increased, setting the size of the kernel to 7.

In this way only the lower side of the top layer of the garment is detected, so from now we look no more for the second, but for the first highest line among the 20 strongest.

This procedure however does not detect the line of interest in a stable way, so we decided to perform three steps:

- detection of the line of interest, as we have just described, without moving for the first 5 frames (see Figure 35),
- at the 5th frame we compute the average θ and ρ parameters in the previous frames and we use those parameters for the successive part,
- from the 6th frame we add memory, starting to perform line tracking instead of line detection.

4.2.4 Computation of the average line

For the first 5 frames, we have decided to remain still and to compute the average of the ρ and θ parameters of the detected lines. This choice is due to the fact that vision is not ideal, so even if we do not move, the parameters of the detected line in each frame change slightly.

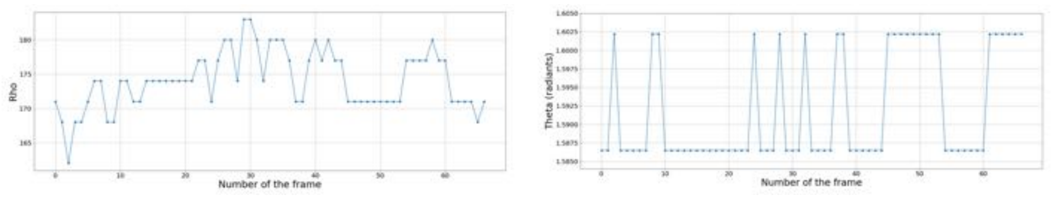

Figure 36: Variation of ρ and θ as the frame changes, camera held in hand, white garment

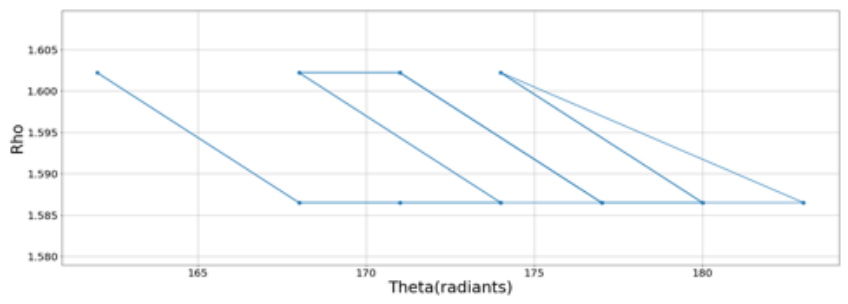

Figure 37: Pairs of ρ, θ parameters, camera held in hand, white garment

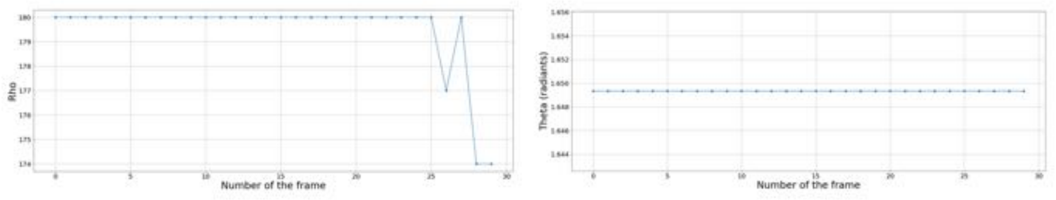

Figure 38: Variation of ρ and θ as the frame changes, camera integrated in the end-effector, white garment

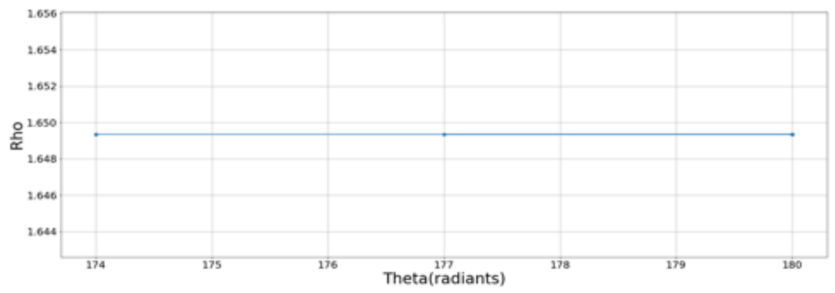

Figure 39: Pairs of ρ, θ parameters, camera integrated in the end-effector, white garment

To verify this assumption, we performed some tests keeping the endoscopic camera in the same position, first holding the camera and then integrating it into the robot's end-effector.

In the first case we expect to have less precision and bigger variations of the parameters, because small oscillations were inevitably produced holding the camera in hand.

Figure 36 shows the graphs of the variations of the ρ and θ parameters as the frame changes when the camera was held in hand, while Figure 37 shows the pairs of ρ, θ parameters. Instead, Figures 38 and 39 show the corresponding plots when the camera has been integrated into the robotic arm.

As we expected, when the camera is integrated into the end-effector the vision is much more stable and there are small variations only in the final frames.

4.2.5 Line tracking

At the 6th frame we use the average θ and ρ parameters as memory and at every new frame:

- if the new position of the line returned by the detection is inside a range of +25 or -25 pixels from the line in memory, then vision is reliable. In this case we update the parameters of the line in memory,

- if the new position of the line returned by the detection is outside the range, then we do not trust vision. In this case we maintain the parameters in memory without updating them and we increase the range where the line can be detected,

- when the range becomes too wide, then the line is considered lost.

Figure 40 shows a case in which the result of vision is accepted (so the line is drawn in green); Figure 41 shows a case in which we do not trust vision, but the range is still sufficiently narrow, so we rely on memory (the line is drawn in yellow) and finally Figure 42 shows a case in which the range has become too wide, so the line is considered lost (and it is drawn in red).

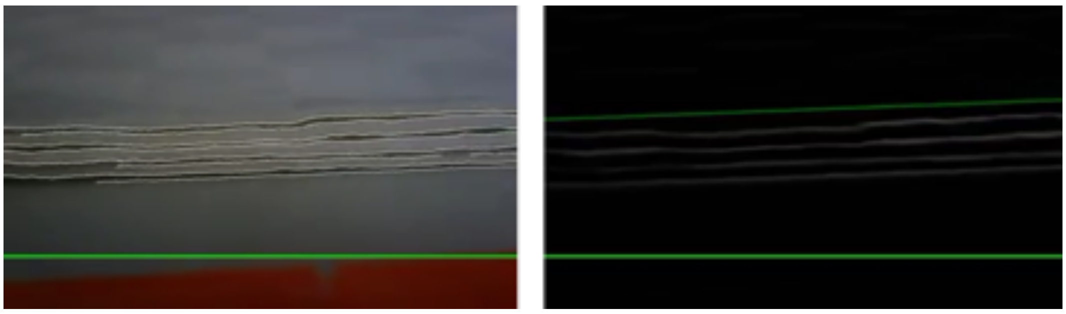

Figure 40: Line tracking, the result of vision is accepted

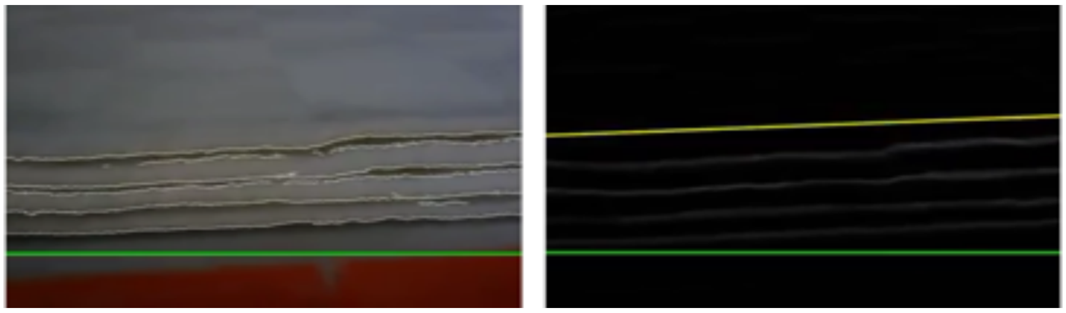

Figure 41: Line tracking, we rely on memory, the range is narrow

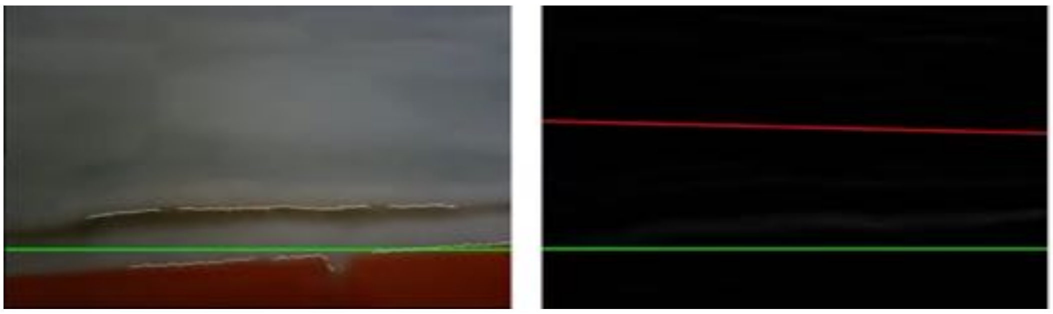
Figure 42: Line tracking, the line has been lost

4.3 Control approach developed in this project

The motion of the end-effector from the pre-grasp to the grasp position is regulated by a a proportional controller. In this case the control rule is not:

$$u(t) = K_P \cdot e(t)$$

where $e(t)$ is the error between the desired and the actual position, but it is:

$$u(t) = K_P(t)$$

because we send constant vectors whose components continue to vary depending on the feedback of the vision.

This controller at each frame sends a message to the whole body controller of the robot, if the distance between the border of the end-effector and the line of interest is greater than zero.

The controller stops sending messages to the whole body controller when the garment hits the camera, so the number of detected lines becomes equal to zero.

The messages contain the new position and orientation that the arm-tool link has to reach in order to approach the garment.

The pose is computed with respect to the reference frame of the base as:

$$\begin{aligned}
&x = old\_x + \delta x\\
&y = 0m\\
&z = old\_z + \delta z\\
&Roll\_angle = 0^\circ\\
&Pitch\_angle = -30^\circ\\
&Yaw\_angle = 0^\circ.
\end{aligned}$$

We are interested in changing only the x and z position, because if the garment is put in front of the camera, changes of the y-coordinate do not provide more information. The increments of the x and z coordinates depend on the relative position of the gripper and the garment; in particular they are set equal to:

$$\begin{aligned}
&\delta x = +0.006m\\
&\delta z = 0m
\end{aligned}$$

when the gripper is at the right height, but it has not reached the garment yet. In this case, in the image the line of interest is between the line y=150 (that is used to filter out the lines that belong to the background) and the edge of the gripper. This motion entails that the lines in the next frame will appear lower.

Otherwise when the gripper is too low and the lines in the image are over the line y=150, the variations are set to:

$$\begin{aligned}
&\delta x = +0.002m\\
&\delta z = +0.001m.
\end{aligned}$$

As in the previous case, after this motion the lines in the image will be lower.

Finally if the gripper is too high, so the lines in the image are below the line that corresponds to the gripper, we need to decrease the height of the arm-tool link, therefore we set:

$$\begin{aligned}
&\delta x = 0m\\
&\delta z = -0.002m.
\end{aligned}$$

This movement entails that the lines in the next frame will appear higher.

This last condition however has to be implemented in another way, because when the line of interest in lower than the line of the gripper, no lines are detected in the image and the algorithm stops.

Therefore we need to decrease the height before the line of interest disappears. To do so we introduce a threshold, called *end_range*, close to the gripper level but higher and whenever the line of interest becomes lower than this threshold we decrease the z coordinate.

However the line may be not perfectly horizontal, so we do not want to decrease the z-coordinate when one of its extremes is lower than the threshold, because if the line is diagonal this would imply that the gripper would be lowered too soon (potentially colliding with the table).

So we perform an attention task: since for the grasping we are interested only in the central part of the line, we focus on the segment in the interval:

$$x \in \left[\frac{image.columns}{2} - 10, \frac{image.columns}{2} + 10\right]$$

When one of the two extremes of this segment is lower than the threshold, then we decrease the z-coordinate. Figure 43 schematizes how the thresholds have been fixed.

We chose to vary the coordinates of only a few millimeters at each frame because the frames change very fast, so with greater values we would probably lose the line before reaching the garment.

Moreover we chose not to change the orientation but only the position because we are very close to the object, so small changes in orientation cause big changes in the image, while small changes in position do not cause such a significant change in the image.

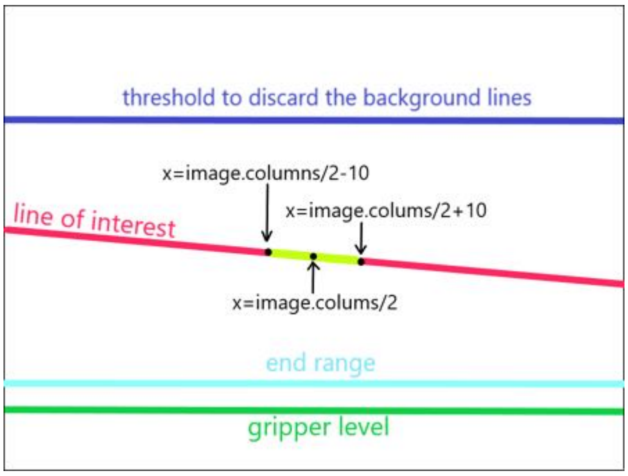

Figure 43: Thresholds in the image

4.3.1 Simulations

Before testing the controller with the robot, we performed some simulations with Gazebo simulator. First of all we obtained the pose to be sent to the whole body controller in order to have TIAGo in the pre-grasp pose shown in Figure 44. With this pose the *arm_tool_link* frame is placed $1m$ away from the *base_frame* in the z direction, $0.5m$ in the x direction, with a pitch angle of 30°.

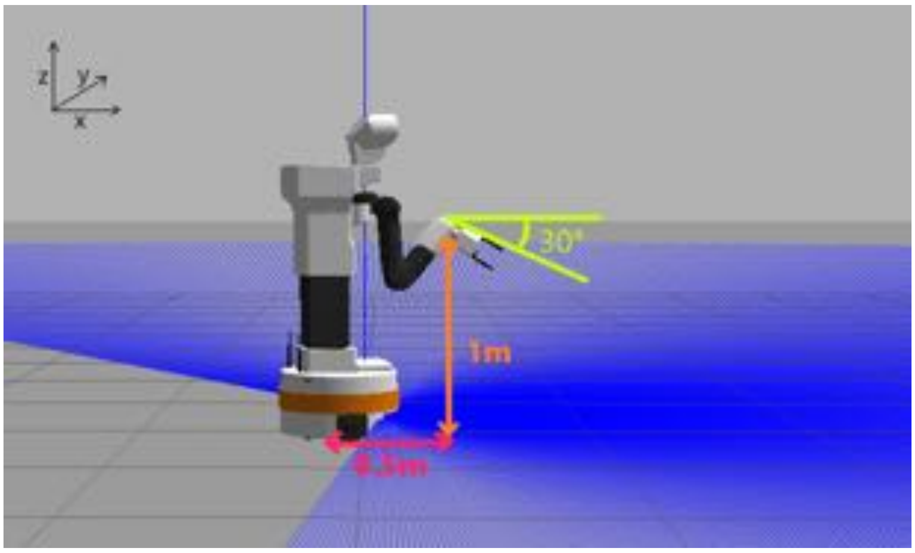

Figure 44: Desired initial position of the arm

We take a pitch angle of 30° because we want to approach the garment proceeding not parallel to the plane, but with the arm inclined towards the table. In this way we exclude part of the background, facilitating the initial detection of the line of interest.

Considering how we designed the controller, once the line of interest has been detected, the robot arm should move towards the garment (increasing the x coordinate of the pose of the *arm_tool_link* frame) and modify its height (increasing, keeping constant or decreasing the z coordinate of the pose) at each new frame.

From the figures 45, 46, we can notice as, getting closer to the garment, the robotic arm stretches to reach it.The right part of these figures shows the simulation, while the left part shows on the upper part the detection of the line of interest and on the bottom part the source image with the Canny edges drawn on it.

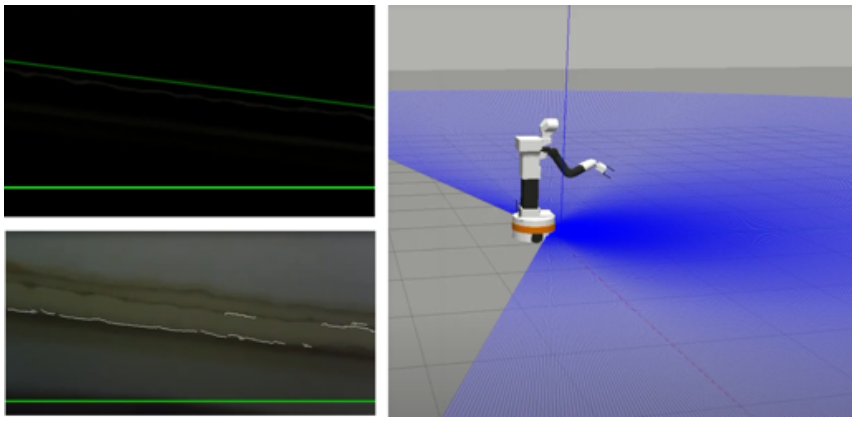

Figure 45: Simulation of the motion regulated by the controller

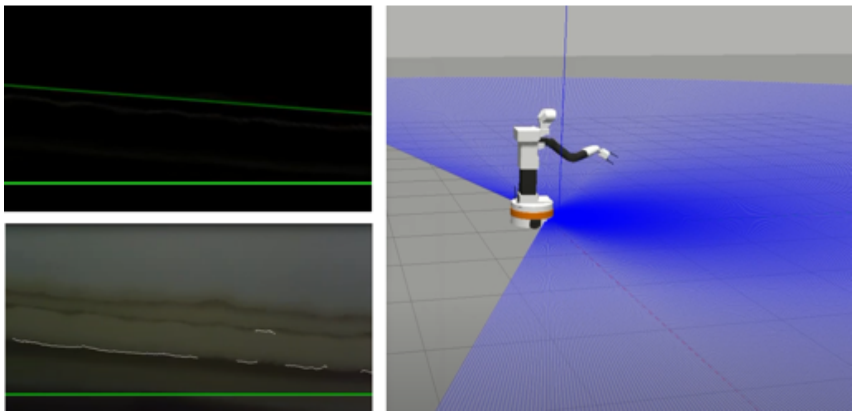

Figure 46: Simulation of the motion regulated by the controller

4.3.2 Messages sent by the controller

As we mentioned before, the controller sends to the Whole Body Control a geometry message containing the desired position and orientation of the *arm_tool_link* reference frame. The orientation is always fixed, as well as the y coordinate of the position.

Instead the x and z coordinates of the position vary, therefore we can represent them as vectors that:

- start from the point (x_i, z_i) corresponding to the coordinates (x, z) sent by the controller at the $i - 1^{th}$ frame,

- end at the point (x_{i+1}, z_{i+1}) corresponding to the (x, z) coordinates sent by the controller at the i^{th} frame.

We expect to see the x-coordinate increasing or remaining constant, while the z-coordinate should increase, remain constant or decrease depending on the camera position with respect to the garment.

In particular, considering the small motions performed, the case when the z-coordinate is increased should never be reached, so we assume to have a behaviour similar to the one depicted in Figure 47.

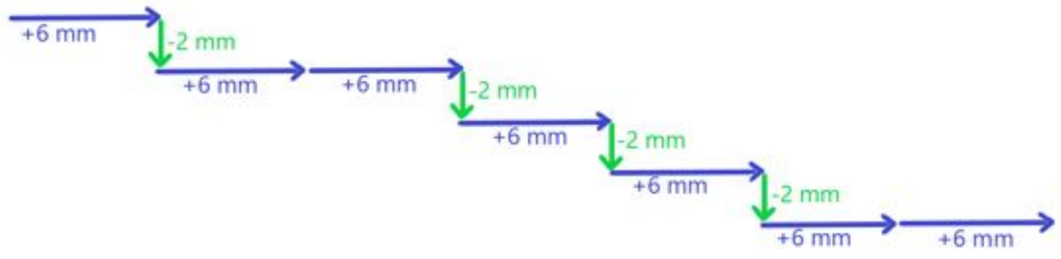

Figure 47: Sequence of (x,z) coordinates sent by the controller

4.3.3 Trajectory of the arm-tool link

Since the Whole Body Controller receives a new position before the arm-tool link has reached the previous one, the resulting movement of the link will be softer and similar to the red trajectory represented in Figure 48.

So in the end we have a trajectory that can be approximated to a diagonal line with the x -coordinate that increases and the z-coordinate that decreases.

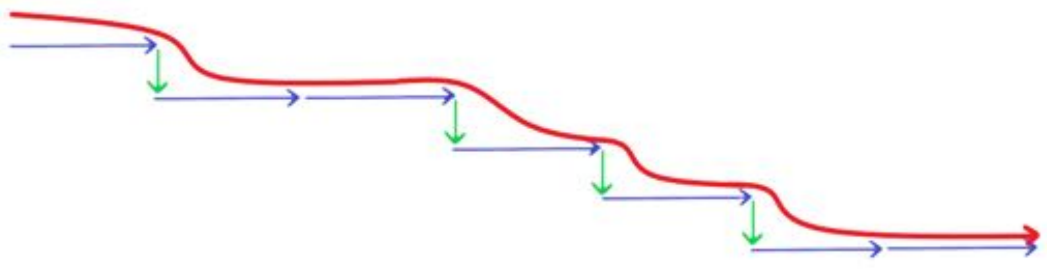

Figure 48: Aspect of the trajectory of the arm-tool link

5 Experimental validation

After having developed this method for visually guided grasping, we now provide its experimental validation. In particular we first describe some improvements made to the approach after accessing real world and robot characteristics and limitations. Then we analyze the stability of the robot and of the controller, namely their ability to maintain a certain position and to move on straight lines or to reach the positions sent by the controller. Subsequently we evaluate the stability of vision at different distances from the garment and under different light conditions. Then we assess the stability and the repeatability of the whole system to changes of the garments, of the surface, of the number of layers and of the initial position of the garment on the surface. Finally we provide an analysis of the average performances of the procedure.

5.1 Real robot specifics and improvement of the approach

In this section we analyze some limitations of the real robot and of the setup in which the tests have been performed and we describe some improvements made to our approach. Specifically: we introduced some pauses useful to reduce the pace in which command are sent to the robot controller in order to avoid saturation; we modified the initial configuration of the robotic arm to allow better manipulability; we studied the best position of the camera on the end-effector; and we developed a stopping condition.

5.1.1 Errors between real and desired position of the link

Initial experiments with the robot revealed that the WBC is not able to reach the positions sent by the controller, this results in the presence of an error between the position of the arm-tool link sent to the WBC, and the real one. However we are interested in exploiting the WBC because it computes the inverse kinematics automatically ensuring joint limit avoidance and preventing self-collisions. Figure 49 shows on the left an example of the variation of the position of the arm-tool link with respect to the base and the coordinates sent to the whole body controller during a test, while on the right there are the relative errors.

We can see that the error on the z coordinate is in the range of 4mm - 6mm. Considering the small distance between the gripper and the garment, a few millimetres can convert a potential success in a fail, so this error is too high.

To have a more precise idea of the uncertainty caused by this error, we tried to send a sequence of messages to the WBC increasing only the x coordinate. In an ideal case we expect to see the arm following a straight line, while the height of the link arm-tool link should remain unchanged.

However, as we can see from Figure 50, there is not only an error on the x coordinate, but also the z-coordinate oscillates considerably. The oscillation of the height of the arm-tool link leads to an unpredictable behavior of the tip, which is not acceptable since we want to perform a fine grasp.

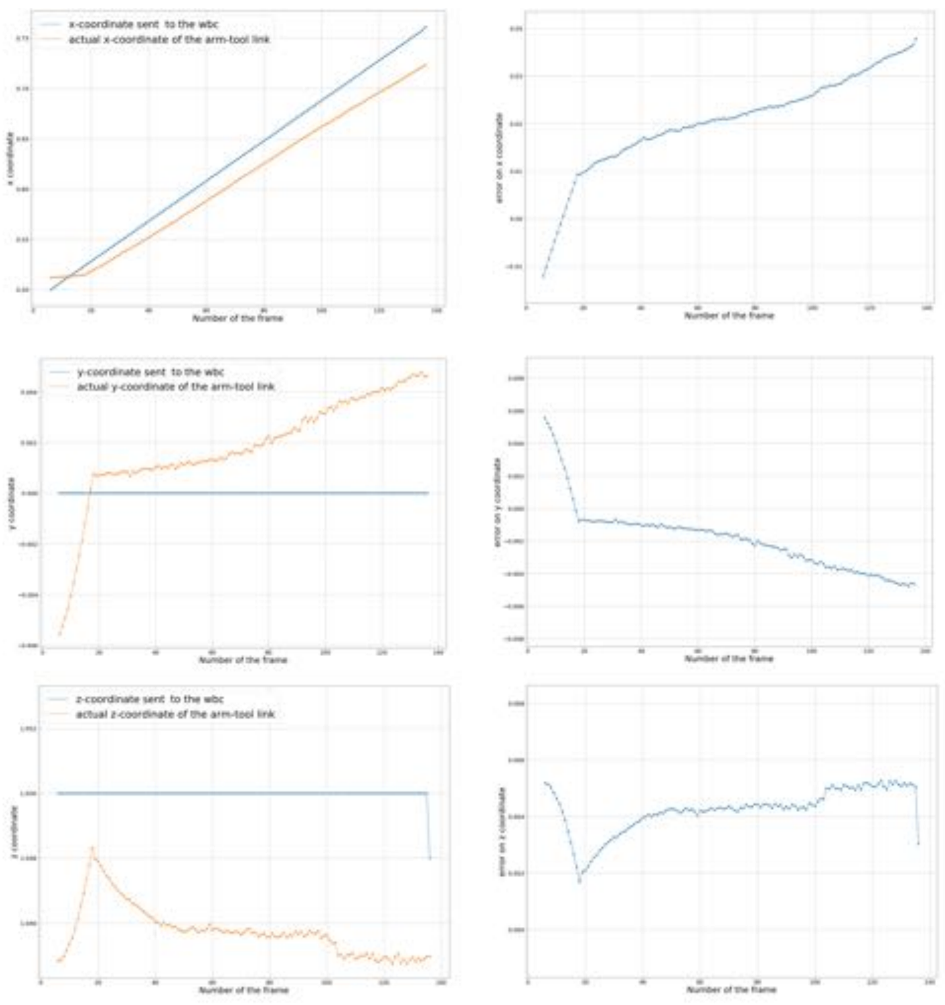

Figure 49: On the left: variation of the position of the arm-tool link and coordinates sent by the WBC, on the right: relative errors

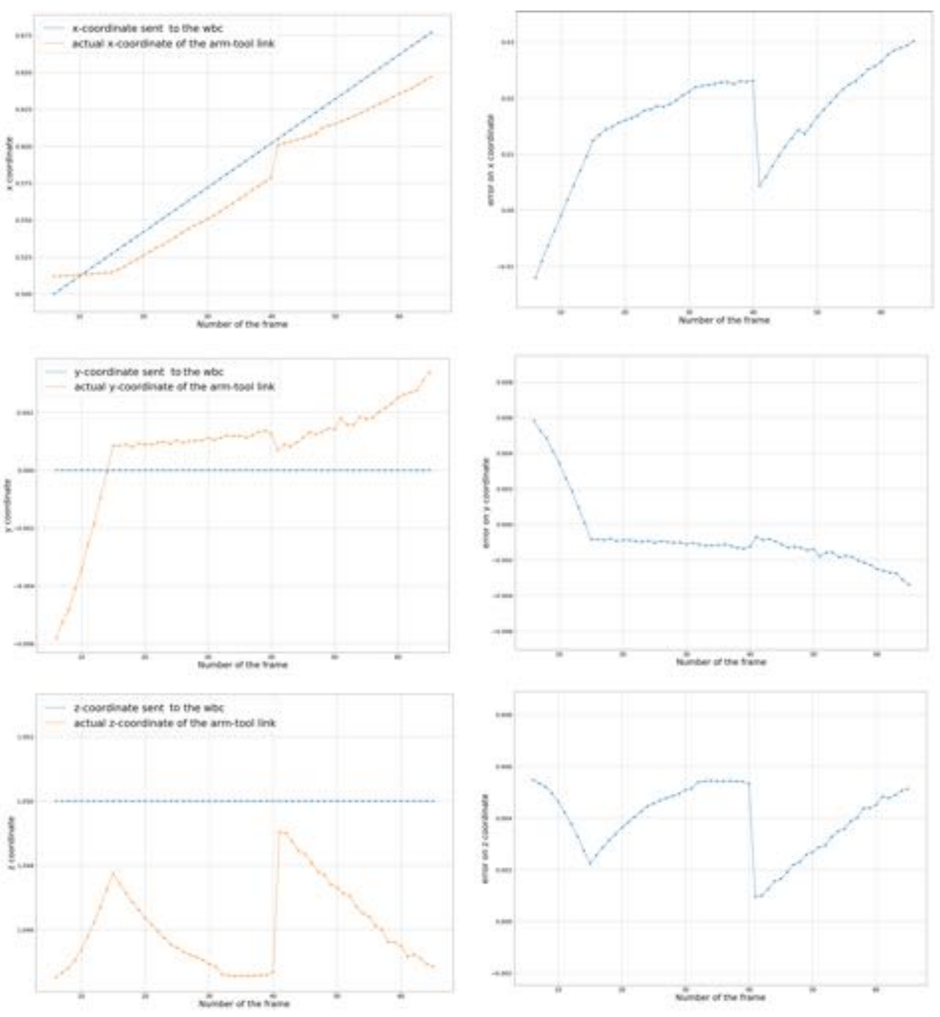

Figure 50: On the left: variation of the position of the arm-tool link and coordinates sent by the WBC, on the right: relative errors

5.1.2 Reduction of the control frequency

To reduce the errors between the x and z coordinates sent by the Whole Body Controller and the real ones, we reduced the number of control interactions with the WBC by introducing a pause before the acquisition of a new frame, immediately after a message is sent to the controller.

The reason of this choice is that if the WBC receives messages too rapidly, the arm-tool link would never be able to reach the position before a new one is commanded. Therefore we performed some tests with the arm and the garment in the same initial position, changing the length of the pause.

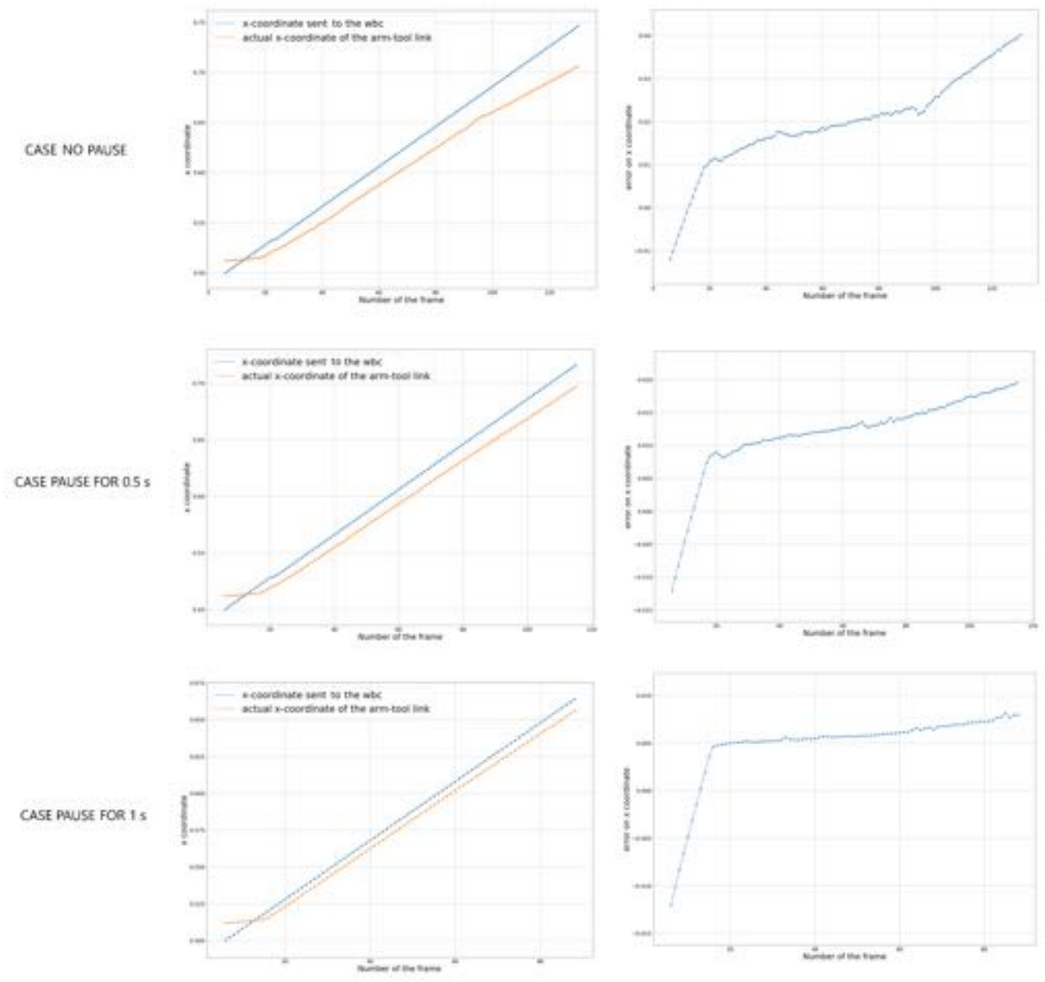

Figure 51: Comparison between the errors on x-coordinate with no pauses, with pauses of half a second and of a second

Figure 51 shows a comparison between the errors on x-coordinate with no pauses, with pauses of half a second and of a second; Figure 52 is analogous for the z-coordinate. As we can see from these figures the bigger is the pause interval, the smaller is the error, but choosing a pause of 1 second will lead to a motion too slow and actually not useful. Therefore we decided experimentally that a pause of maximum 0.5 seconds was the proper compromise.

We tried to perform some tests with 0.5 seconds of pause, but starting from the same initial configuration we obtained some successes (as shown in Figure 53), but also fails, for example in some cases the gripper missed the layer or it inserted under the second layer, or in other cases it hit the upper layer pushing it (as shown in Figure 54). This lack of repeatability is deleterious for our purposes and it is due to the fact that the Whole Body Controller, when receives a message, can use all the 7 degrees of freedom of the arm and the torso to achieve the position with the arm tool link in the best way possible.

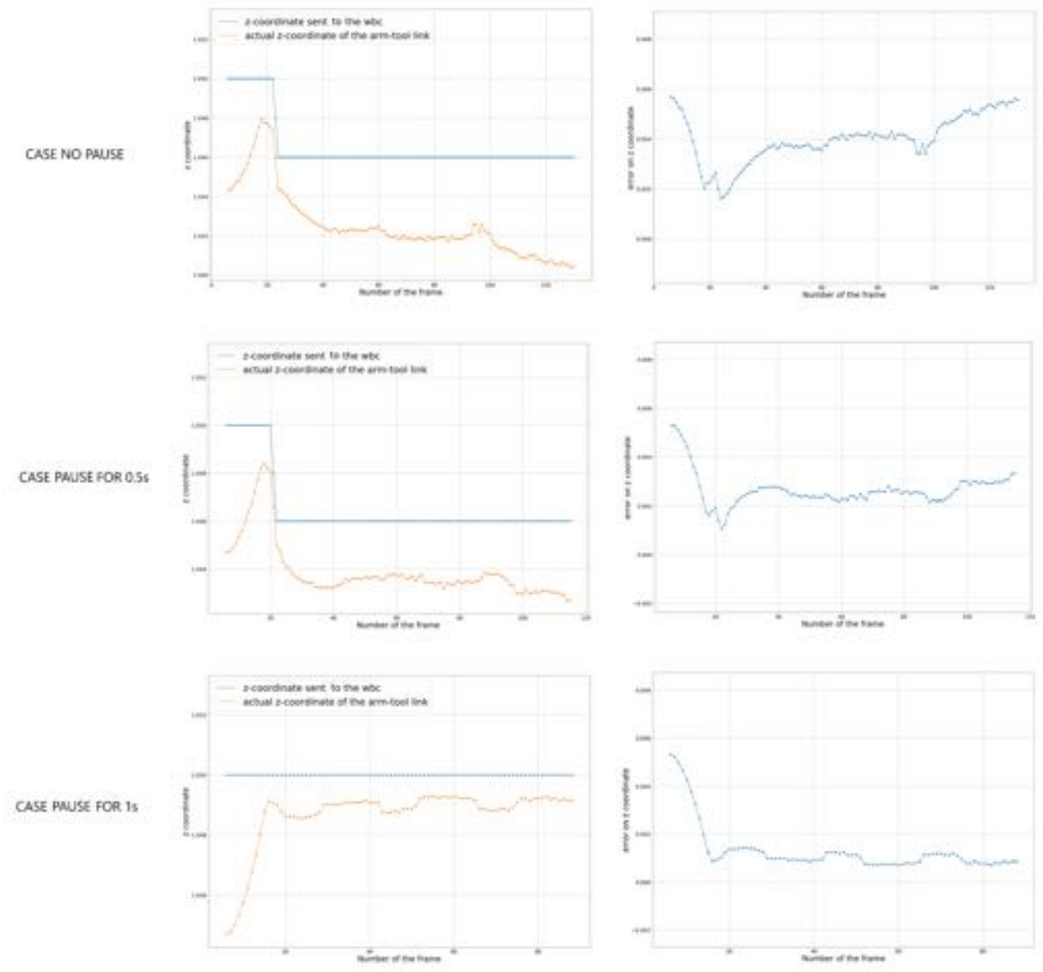

Figure 52: Comparison between the errors on z-coordinate with no pauses, with pauses of half a second and of a second

The kind of approaching motion is always similar. We observed that the initial position that was used may not coincide with the most natural motion, causing complex control actions resulting in the arm doing movements that cause unwanted oscillations of the gripper. This issue is addressed in next section.

5.1.3 Change of the initial position of the arm

In order to have a better control of the movement of the arm we decided to change the initial configuration of robot TIAGo. In particular, we imposed as

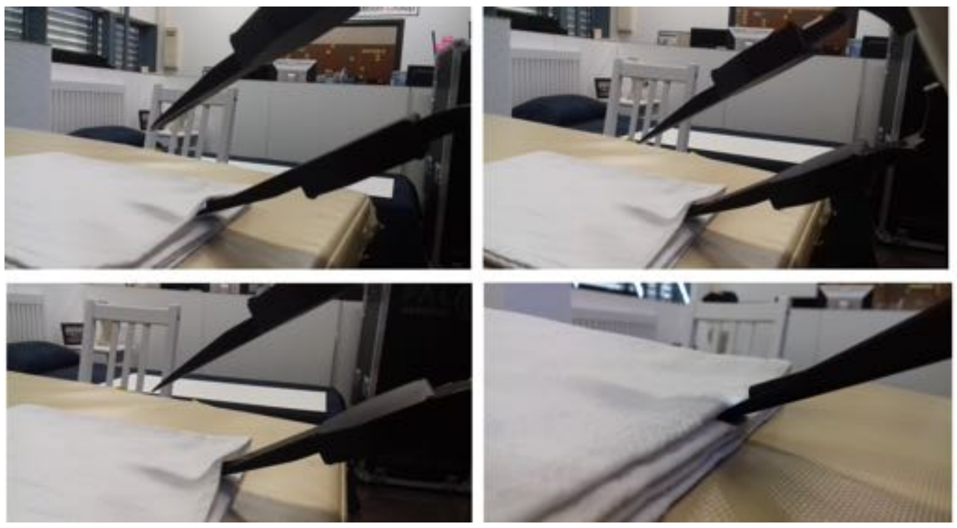

Figure 53: Cases of success starting from the same initial configuration

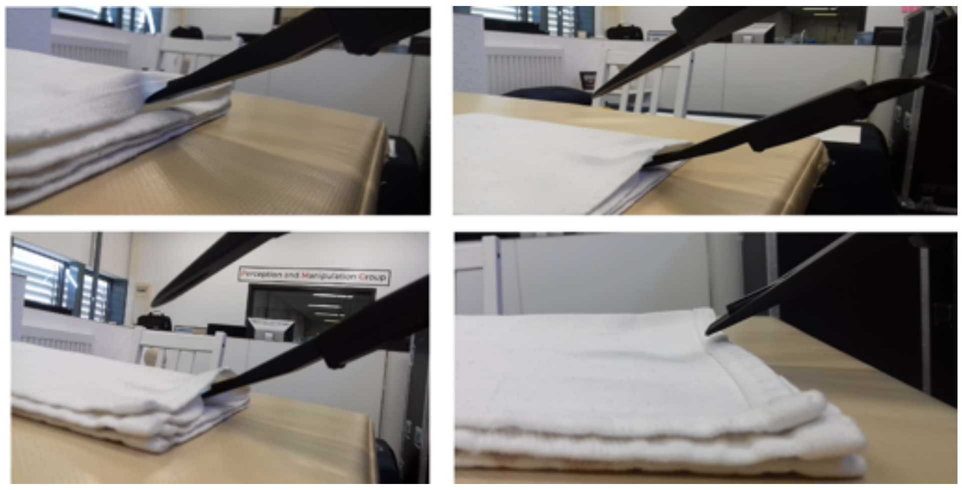

Figure 54: Cases of failure starting from the same initial configuration

initial position of the arm-tool link no longer:

$$x = 0.5m,$$
$$y = 0m,$$
$$z = 1m$$

as before, but:

$$x = 0.75m,$$
$$y = 0m,$$
$$z = 0.75m.$$

Therefore the robot has the arm lower and more extended than before. Figures 55 and 56 show the difference between the two configurations.

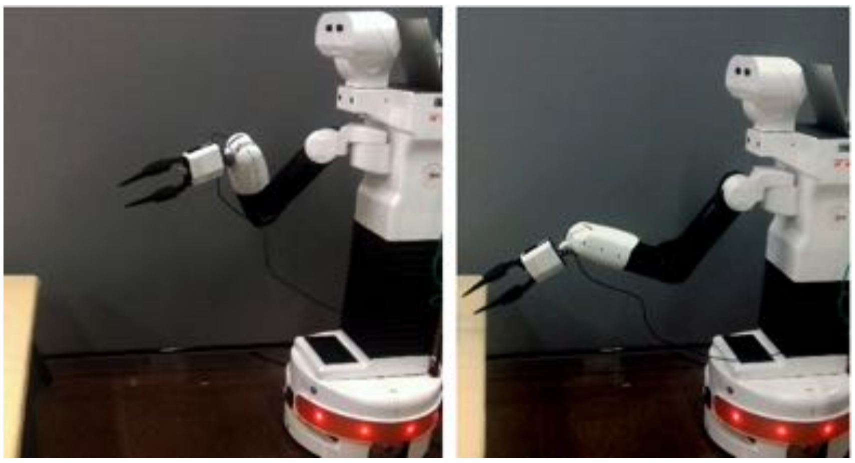

Figure 55: Two initial configurations of TIAGo

With the old configuration, to approach the garment the robot mainly exploited the 2nd, 4th and 5th arm joints, while the torso remained fully extended. However, the rotations of the arm joints caused unnatural movements that increased the variation of the height of the gripper.

The advantage of the new configuration is that TIAGo, to approach the garment, extends the arm using the 4th joint of the arm and changes the height exploiting the mobility of the torso instead of rotating the other joints. We are therefore trying to limit the number of degrees of freedom involved in the movement, in order to have more predictable and precise movements. Figures 57 and 58 report the x and z behaviours and errors with the new configuration. As we can notice from these figures, the error on the x-coordinate is of 1cm after 100 frames, while with the previous configuration it was around 1.8cm. Analogously also the error on the z-coordinate decreases, indeed after 100 frames it is about 1mm, while before it was more or less 3mm.

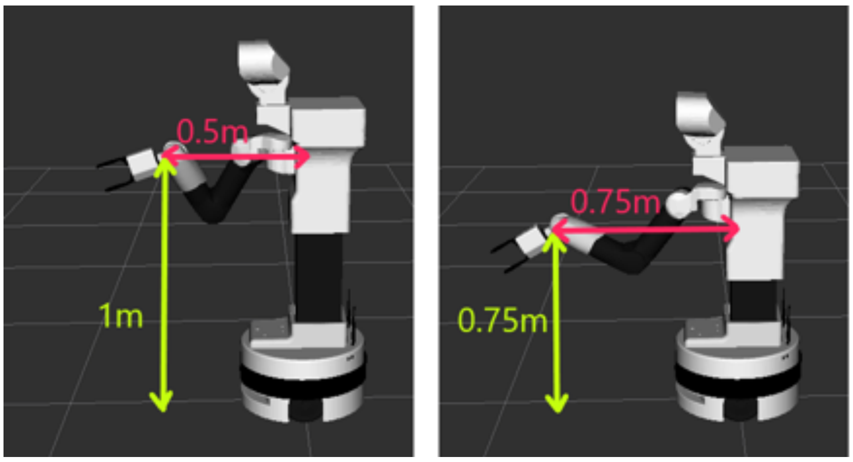

Figure 56: rviz schemes of the two initial configurations of TIAGo

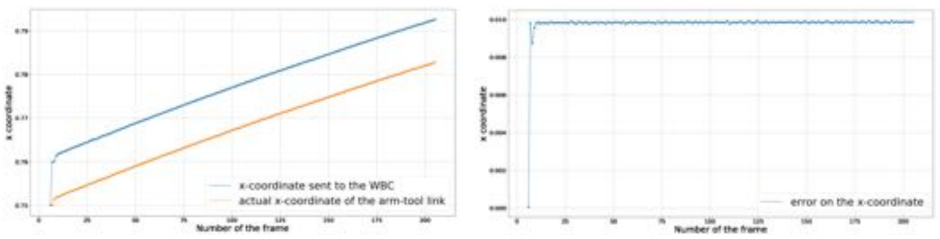

Figure 57: Behaviour of the x-coordinate and relative error, with the new configuration

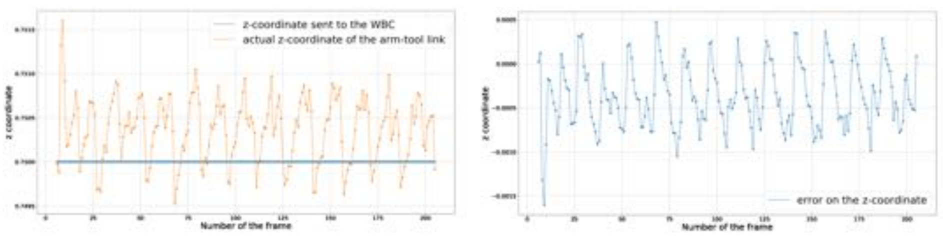

Figure 58: Behaviour of the z-coordinate and relative error, with the new configuration

5.1.4 Change of the camera position

The first attempt to solve the task was to use the camera that was already integrated into the one of the gripper fingers (see Figure 59 left). However, a problem that emerged from the tests was the loss of the line when the gripper was really close to the garment.

Indeed, even with the introduction of the threshold *end_range*, sometimes the line of interest was covered by the line of the gripper, so it was lost before being able to sufficiently decrease the z-coordinate. Consequently the algorithm stopped too early because no lines were detected.

To avoid this issue it has been necessary to change the position of the camera. We placed the camera further away, no longer integrated in the gripper but attached externally to the end-effector, so that the line of interest was not lost even when it became lower than the line of the gripper. Figure 59 shows the difference between the old and new position of the camera.

With the camera in this new position the algorithm described before remained almost unchanged, because, as we can see from Figure 60, the endoscopic camera still sees the tip of the gripper.

Only two things changed in the algorithm:

- we no longer need to use the threshold *end_range*,
- we had to define a new stopping condition, because the camera will no longer be hit by the garment in case of success, so the number of detected lines will always be greater than zero.

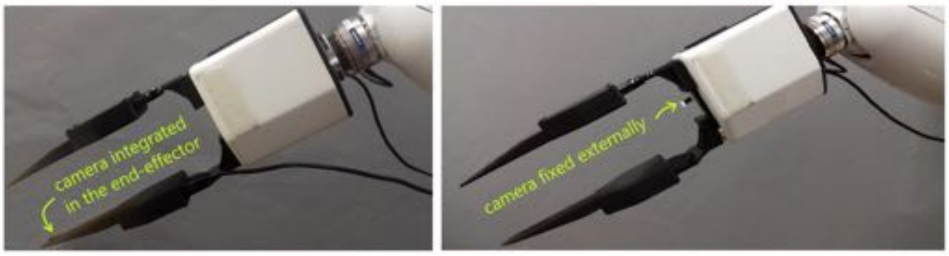

Figure 59: Two different positions of the camera

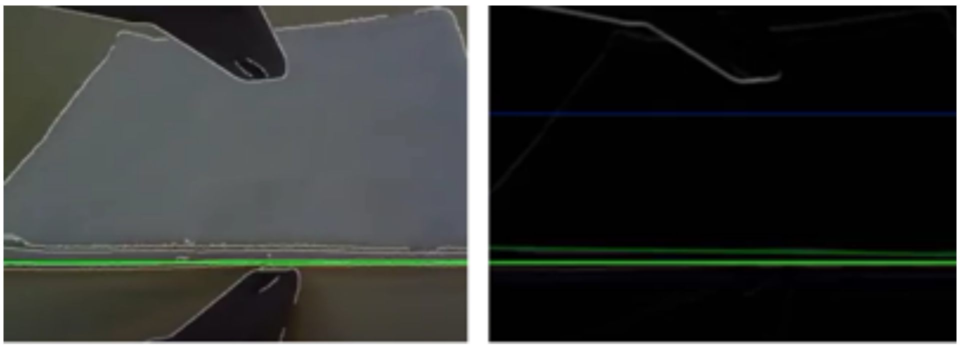

Figure 60: Type of image coming from the camera in the new position (left), line of interest and line corresponding to the gripper (right)

5.1.5 New stopping condition

For the new stopping condition the black color of the end-effector has been exploited. The idea is that in the image the tip of the end-effector appears fixed, so it always corresponds to the same subset of pixels. Therefore if we locate the region of the image corresponding to the tip, this coincides with a subset of pixels whose average RGB color components are close to zero; indeed, the RGB components $(0, 0, 0)$ correspond to black, while $(255, 255, 255)$ correspond to white.

The region of the tip, as we can see from Figure 61, has been identified as the set of pixels within the triangle delimited by the lines:

$$\begin{cases} x = -4y + 445 \cdot 4, \\ x = \frac{15}{10}y - \frac{1245}{10}, \\ y = 360. \end{cases}$$

So we computed the average value of the RGB components of the pixels within this region and then we calculated the sum of these values.

When the end-effector is in the final position just before the closing of the gripper, the garment covers the tip, therefore the average RGB color components of the pixels in the tip region increase. In particular with the white garment this peak will be particularly pronounced.

To stop the movement it is therefore sufficient to fix a threshold and to end the motion as soon as the sum of the average RGB components of the pixels in the region exceeds this value.

The only drawback of this technique is that we cannot use garments black or too dark, because otherwise this threshold is never exceeded.

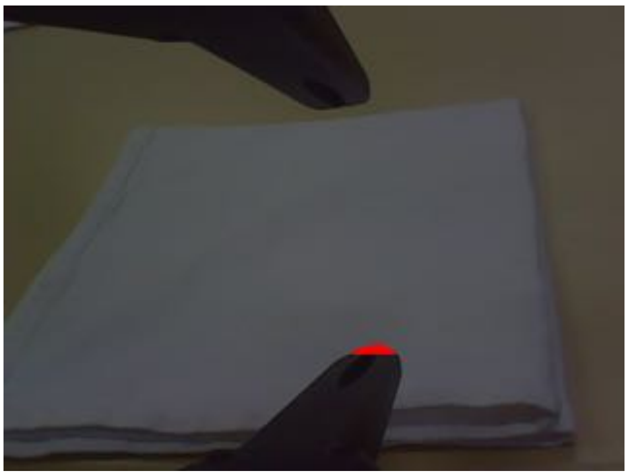

Figure 61: Region of pixels corresponding to the tip of the end-effector

5.2 Analysis of the stability of robot and controller

In this section want to evaluate the stability of the robot, namely its ability to keep the arm in the controlled position.

5.2.1 Robot still

To do this we derived the position of the arm-tool link keeping the robot stationary for N=1000 frames. In an ideal case the x and z coordinates should remain perfectly constant for all frames, but of course in a real case there are micro-oscillations of the robot that cause variations of the coordinates.

Figures 62 and 63 show the variations of the x and z-coordinate of the position of the arm-tool link.

From these figures we can see that even when it receives no messages the controller does not keep the arm perfectly still, but the errors on the x and z-coordinates are really small, of the order of $10^{-4}m$.

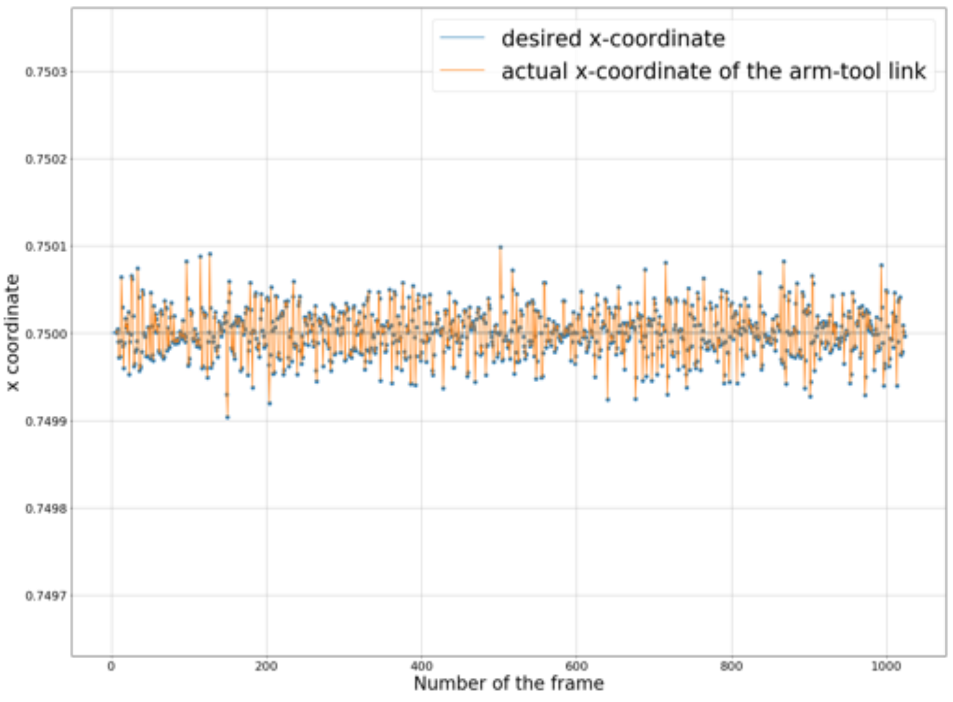

Figure 62: Variation of the x-coordinate of the arm-tool link

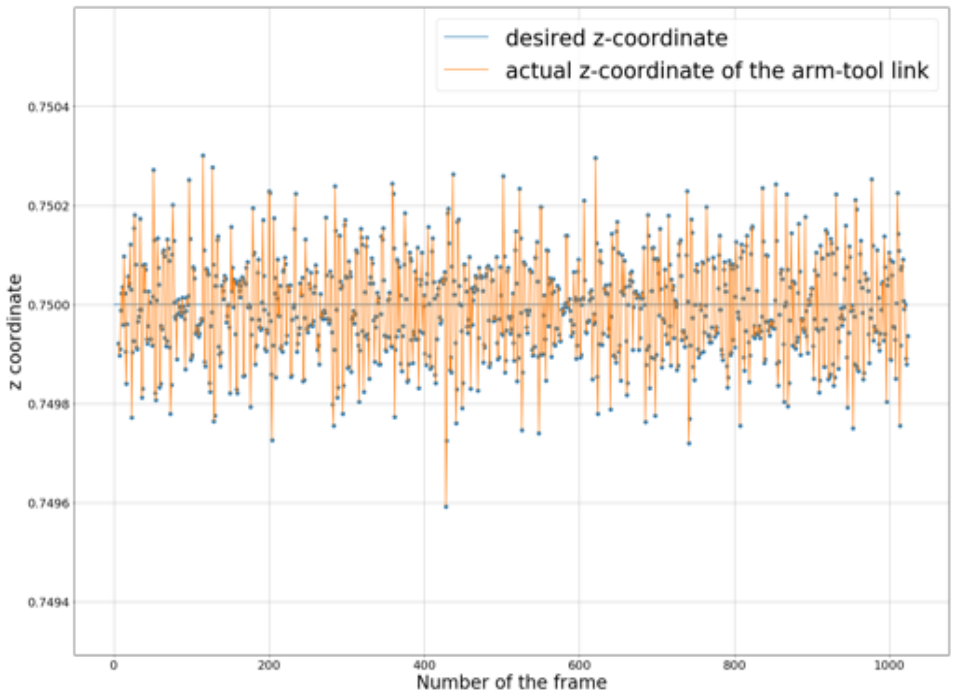

Figure 63: Variation of the z-coordinate of the arm-tool link

However in our case we do not keep the robot perfectly still for a long time, because at each frame the WBC receives a new position to be reached.

Therefore we want to evaluate the ability of the robot to avoid oscillations after each movement. To do so we analyzed the behaviour of the robot when it is commanded to move on a straight line.

5.2.2 Follow an horizontal line

First of all we tried to follow an horizontal line, so we sent a constant value to z (z=0.75m as in the initial position) and at each frame we increased the x coordinate of 1 cm.

As we can see from Figure 64 the arm-tool link is not able to reach the value of the x-coordinate sent by the whole body controller and we have an error around 9mm. This is due to the fact that the gains of the Whole Body Controller are set very low to ensure safety in case of human-robot interaction. When the arm starts moving in the commanded direction, the whole body controller receives a new message before the arm is able to reach the position.

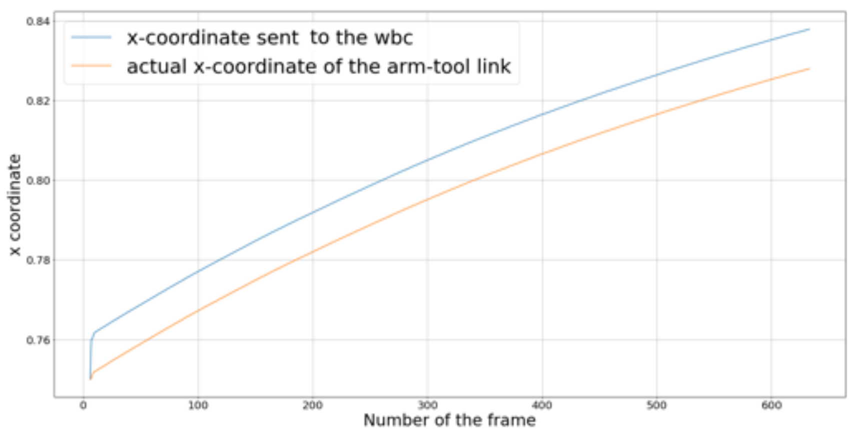

Figure 64: Variation of the x-coordinate of the arm-tool link, when we send a constant increment to the x-coordinate and a fixed reference value to the z-coordinate

The z-coordinate, on the other hand, should remain fixed at 0.75m, but as we can see from Figure 65, it oscillates with an error of about 1mm.

However in our procedure, to be as more general as possible, when we increase the x we do not send a fixed value of z but we send the current position of z, which in theory should remain constant between two consecutive frames, but actually oscillates, as we have just seen in Figure 65.

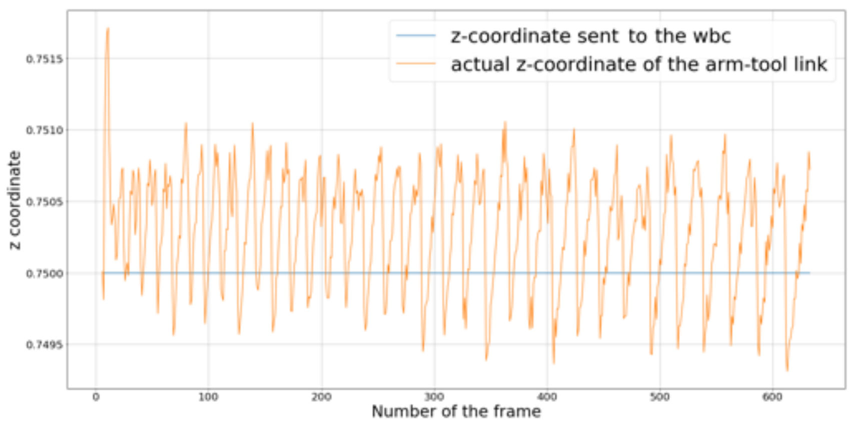

Figure 65: Variation of the z-coordinate of the arm-tool link, when we send a constant increment to the x-coordinate and a fixed reference value to the z-coordinate

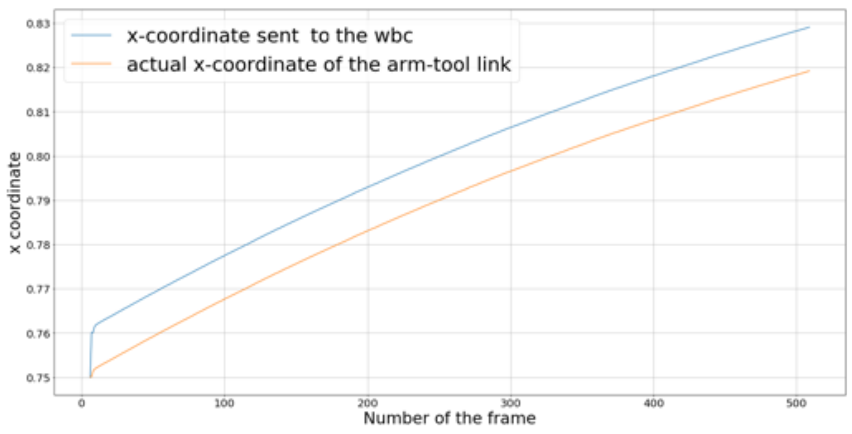

Figure 66: Variation of the x-coordinate of the arm-tool link, when we send a constant increment to the x-coordinate and we read the current value of the z-coordinate

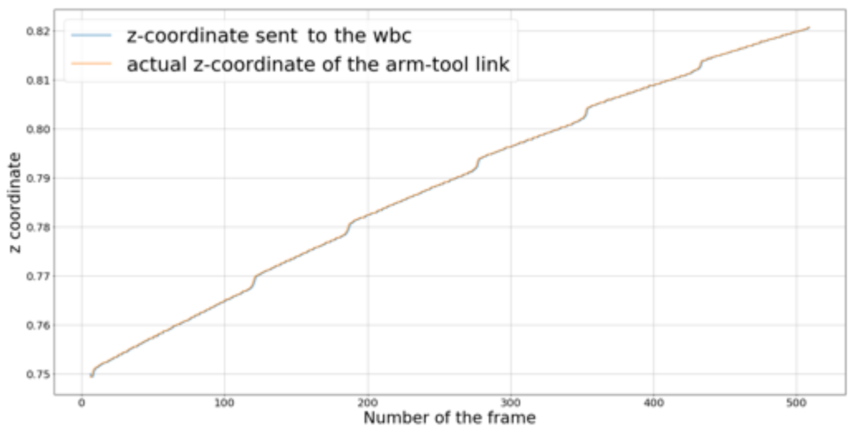

Figure 67: Variation of the z-coordinate of the arm-tool link, when we send a constant increment to the x-coordinate and we read the current value of the z-coordinate

Figures 66 and 67 shows the variation of x and z when we constantly increase x and we update z to the current value.

From these figures we notice that sending the current z-coordinate instead of a fix value, we integrate the error between the initial z-coordinate of the arm-tool link and the desired one, causing an undesired increment of z-coordinate. In this experiment, for example, in 500 frames it increases of 7cm.

5.2.3 Follow a vertical line

Similarly, if we try to make a vertical movement of the arm-tool link, decreasing the z-coordinate of 2mm at each frame and keeping the x-coordinate to the fixed value of 0.75m, we get an error also on the x-coordinate. Figures 68 and 69 show the behaviour of x and z coordinates respectively.

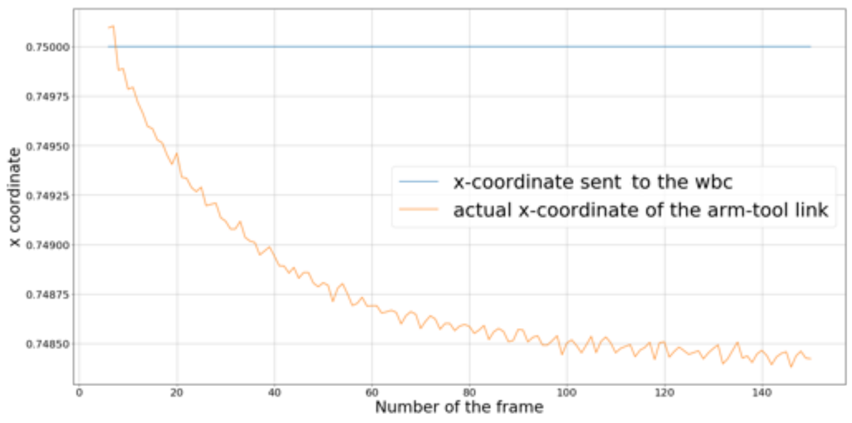

Figure 68: Variation of the x-coordinate of the arm-tool link, when we send a fixed reference value to x-coordinate and we constantly decrease the z-coordinate

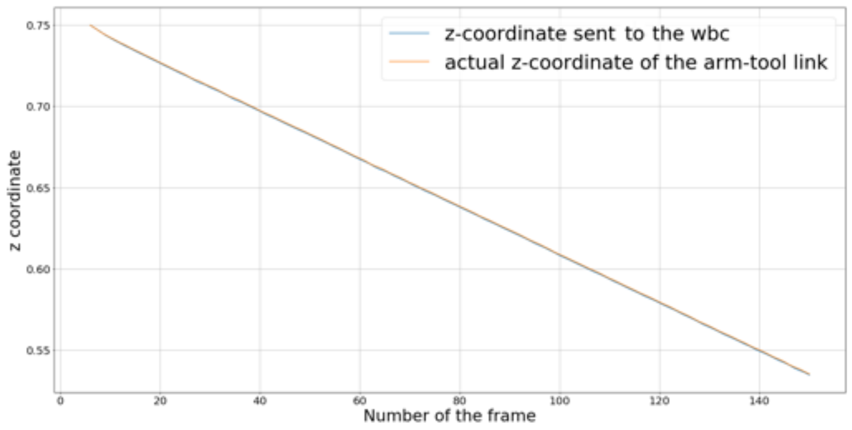

Figure 69: Variation of the z-coordinate of the arm-tool link, when we send a fixed reference value to x-coordinate and we constantly decrease the z-coordinate

We can note that the error on the z-coordinate is very small, this means that the WBC is able to follow quite well the instructions received for changes of position in the z-direction, a lot better than changes in the x-direction. This is due to the fact that to change the x position the robot involves the whole

ETSEIB

structure of the arm, while to modify the z-coordinate the WBC can exploit the motion of the torso by performing a simple linear movement.

We can also observe that in this case the error on the x-coordinate does not oscillate around the desired value, but decreases exponentially. Anyway the total error after 150 frames is less than 2mm. However in our procedure when we decrease the z-coordinate we do not keep a constant value for the x-coordinate, but we send the current value.

As we have just seen, the x-coordinate does not remain constant between consecutive frames, but it is always lower than the desired value. This causes an integration of the error between the initial x-coordinate of the link and the desired one, which results in the decrease of x shown in Figure 70. Instead the behaviour of z-coordinate, shown in Figure 71, is good because the reference is followed with a small error.

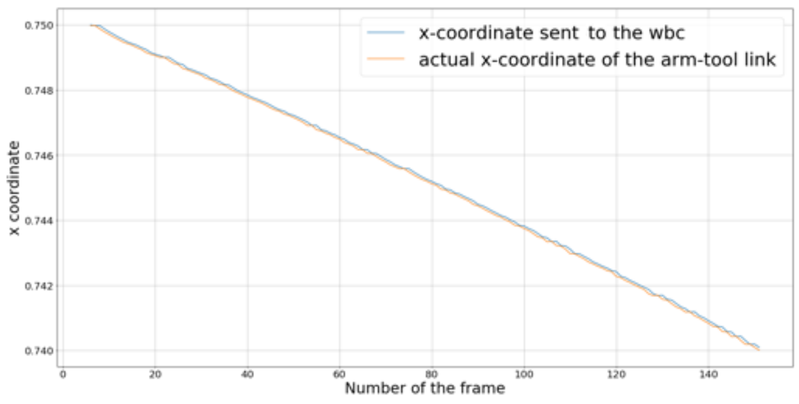

Figure 70: Variation of x-coordinate of the arm-tool link, when we read the actual value of the x-coordinate and constantly decrease the z-coordinate

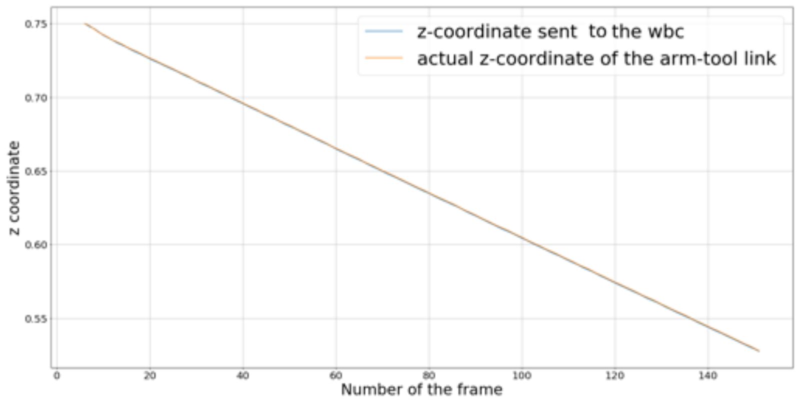

Figure 71: Variation of z-coordinate of the arm-tool link, when we read the actual value of the x-coordinate and constantly decrease the z-coordinate

5.2.4 Correction of the controller

We want to avoid undesired behaviours on the x and z coordinates caused by the integration of the initial errors, because they could cause oscillations during the movement to approach the garment. To do this we modify our controller so that at each frame it sends to the WBC a message like:

$$(x\_memory, actual\_y, z\_memory)$$

where x_memory and z_memory are two variables that are initialized to the x and z coordinates of the arm-tool link in the initial position and then updated to:

$$x\_memory = current\_x\_coordinate\_of\_arm\_tool\_link + \delta x$$

when the increment of the x coordinate is greater than 0, and:

$$z\_memory = current\_z\_coordinate\_of\_arm\_tool\_link + \delta z$$

when the increment of the z-coordinate is lower than 0.

So we update the coordinate that we want to change and we do not read the current value of the other coordinate, but we fix it equal to the last value sent to the WBC. If we now try to send to the WBC only increments of the x coordinate, to check whether it is able to move the arm-tool link in horizontal line, we obtain the behaviours of the x and z coordinates shown in Figure 72. So as in the first case the z coordinate does not diverges, but oscillates of about 1mm around the desired value.

Analogously, when we send to the WBC only decreases of the z-coordinate, to see if it is able to move the arm-tool link in a vertical line, we obtain the behaviours of the x and z coordinates reported in Figure 73. As before the z coordinate follows really well the reference, while the x coordinate decreases.

However, as we can notice this decrease is of about 1.7mm after 100 frames, so considering that during our motion we mostly increase the x coordinate and sometimes we decrease the height, the error that we will have during a normal motion is really small.

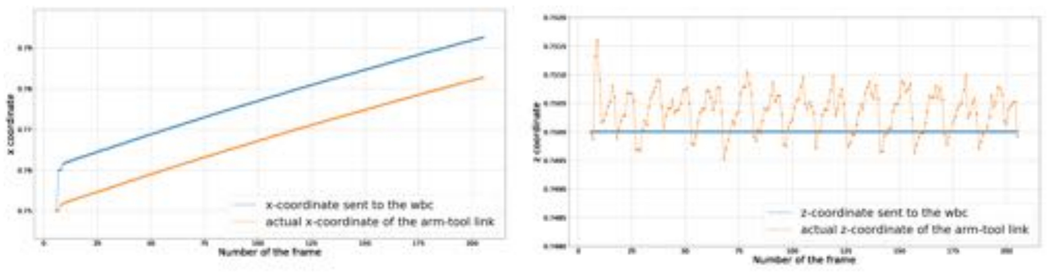

Figure 72: Variation of the x and z-coordinate of arm-tool link, horizontal line

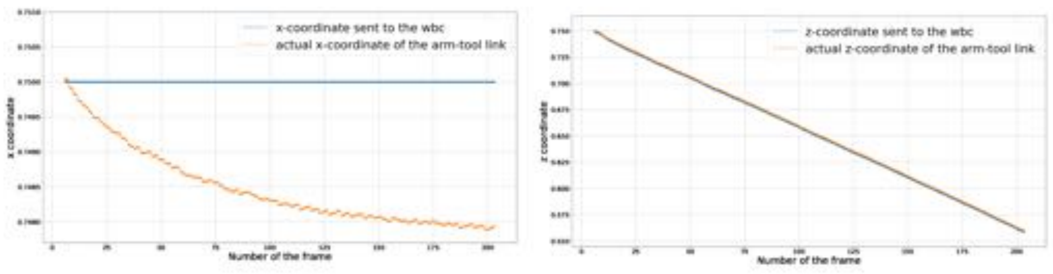

Figure 73: Variation of the x and z-coordinate of arm-tool link, vertical line

5.3 Analysis of the stability of vision

Another aspect that we wanted to assess is the validity of the vision part of our procedure. To do this we observed where the line of interest is perceived when the robot is stationary and we tried to evaluate the sensitivity of the procedure to light variations.

5.3.1 Detection of the line when the robot is still

We evaluated where the line of interest is detected in two cases: when the end-effector is away from the cloth and when it is close, because we noticed that the vision is more imprecise at the beginning of the movement.

End-effector far from the garment
We first detected the line of interest keeping the robot still with the tip of the end-effector at about 15cm from the garment. This can approximate the relative position of the camera and the cloth at the beginning of the motion.

As we can see from Figure 74, which shows the variation of ρ and θ parameters as the number of frames increases, at this distance vision is not very precise.

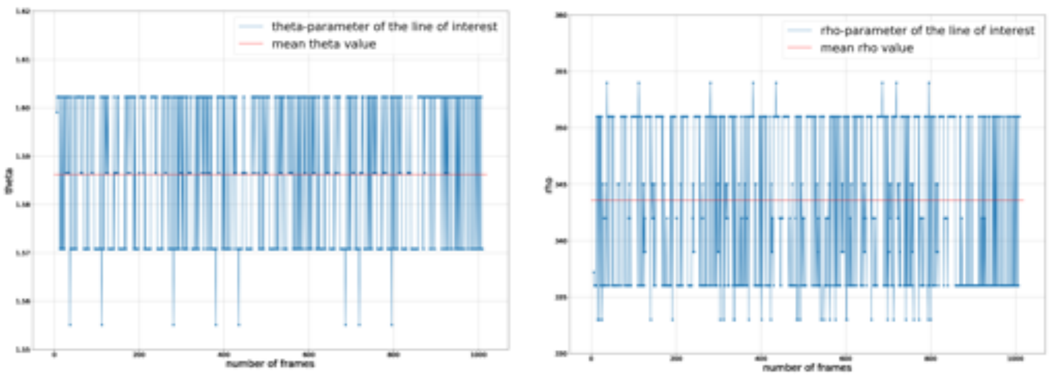

Figure 74: Variation of rho and theta parameters

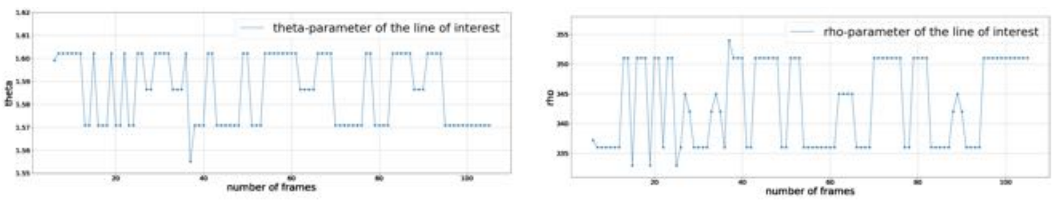

Figure 75: Zoom of the previous graphs

This is consistent because when we are far from the garment the line of interest in the image is only at few pixels from the other lines, so vision may easily mistake another line below for the one of interest.

In particular if we enlarge the graphs taking only the first 100 frames (see Figure 75), we notice that the rho parameter varies in the range $[337, 351]$ pixels, with some peaks at 335 and 353; less than 20 pixels are really a few considering that the image has $640x480$ pixels. Instead θ varies in the range $[1.571, 1.604]rad$, with rare peaks at $1.555rad$. A variation of 0.033 radians corresponds to a variation of 1.89 degrees.

End-effector close to the garment

Then we placed the garment at 2cm from the tip of the end-effector, to simulate the vision when the gripper is about to grasp the garment.

In this case, as we can see from the graphs shown in Figure 76, the vision is much more stable, because in the image the line of interest is more distant from the other lines than in the previous case, so it is easier to detect it correctly.

In particular if we enlarge the graphs taking only the first 100 frames (see Figure 77), we notice that ρ varies in the range $[312, 318]$, while θ is constantly $1.5856rad$, with a few peaks at $1.603rad$ around frame 650.

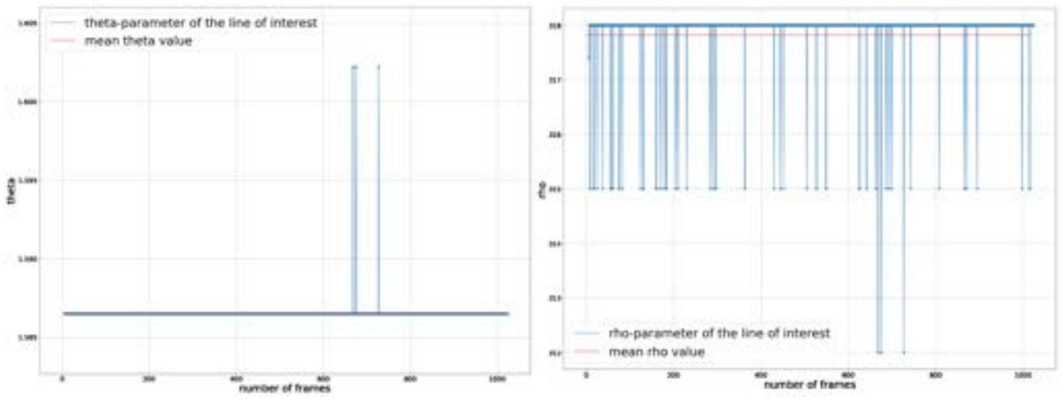

Figure 76: Variations of rho and theta parameters, garment closer

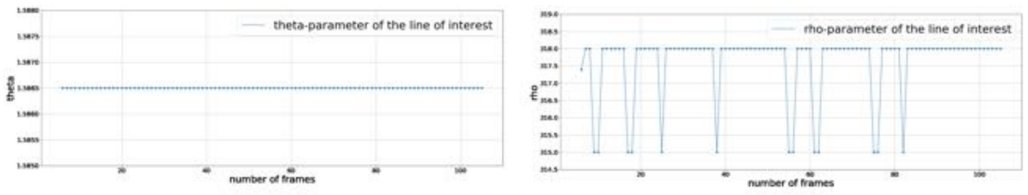

Figure 77: Zoom of the previous graphs

We can therefore conclude that the vision is more stable when we are close to the garment. This is positive because errors of a few millimeters in the last part of the movement can lead to the failure of the test, while errors in the first part of the movement can be corrected in the following part.

5.3.2 Stability to different light conditions

Dealing with vision the lighting is the most delicate point. Indeed the quality of vision can be influenced by the amount of light sensed by the camera: both too much light and too little could cause errors in the extraction of the edges with Canny edge or in the detection of the lines with Hough lines transform leading to the failure of a test.

Therefore we evaluated the stability of vision even with different lighting conditions: while the data in the previous section were obtained with the standard laboratory lighting, which varies between 554 and 560 lux, this section contains data obtained by adding a strong lamp to the left of the robot. With the addition of this light source in the garment area the average lighting is of 825-832 lux.Figure 80 shows the difference between the two light conditions.

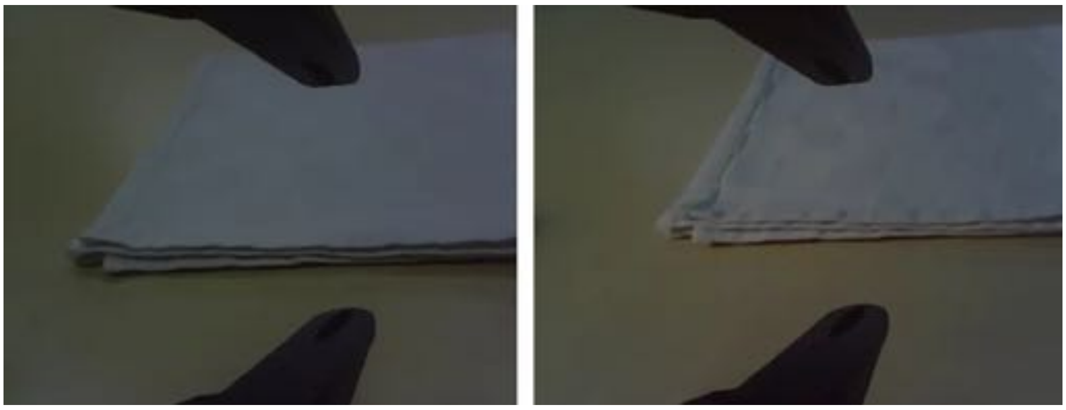

Figure 78: Different light conditions, 554-560lux on the left, 825-832lux on the right

End-effector far from the garment

As before we evaluated the behaviour of vision with the end effector at different distances from the garment. First we detected the line of interest keeping the robot still with the tip of the end-effector at about 15cm from the garment.

As we can see from Figure 79, which shows the variation of ρ and θ parameters as the number of frames increases, at this distance vision is not precise.

In particular, if we look at the zoom of the previous figures (see Figure 80) we can notice that θ oscillates in the range $[1.51, 1.525]$, so a narrower range than before, but it has some very high outliers that reach the value 1.603. Instead ρ oscillates more than before and in a bigger range: $\rho \in [248, 272]$.

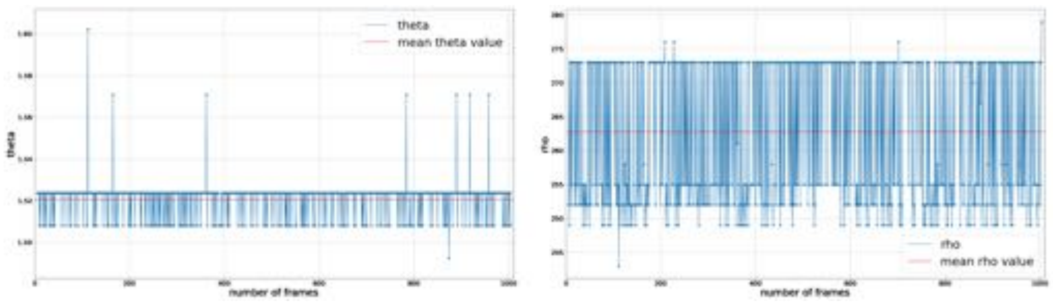

Figure 79: Variation of rho and theta parameters

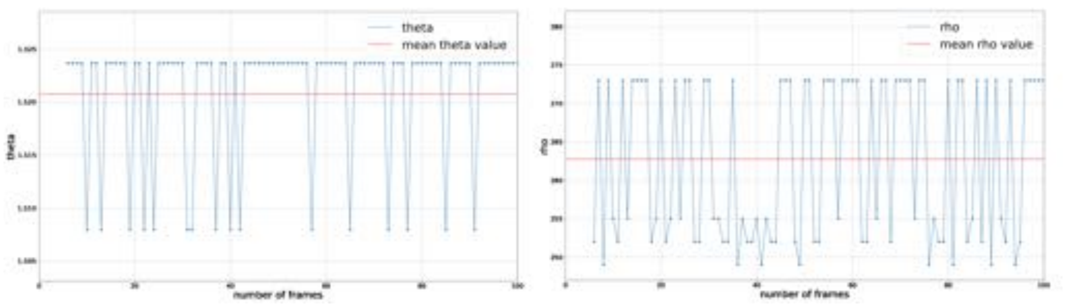

Figure 80: Zoom of previous figure

End-effector close to the garment

Then we placed the garment at 2cm from the tip of the end-effector. In this case, as we can see from Figure 81, vision is more stable, but still worse than in the other lighting condition.

In particular we can see that θ is constantly equal to 1.5551, without outliers, while ρ oscillates in a narrow range ($\rho \in [246, 248]$), but more often than with the previous lighting condition.

We can therefore conclude that the vision is more stable when we are close to the garment and with a lower lightning level.

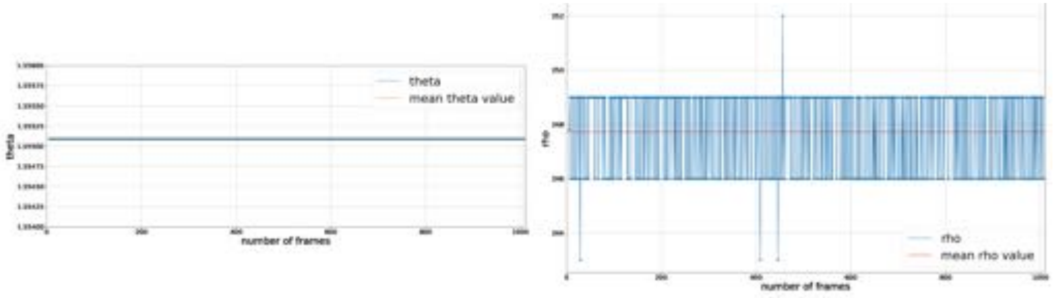

Figure 81: Variations of rho and theta parameters, garment closer

5.4 Analysis of the stability of the whole system

In this section we analyze the stability of the whole system. In particular we assess whether it is stable to changes of garment, of the surface on which the garment is placed, to changes of the number of visible layers, of the initial position of the garment with respect to the camera and to consecutive repetitions.

Figure 82 shows the typical movement of the robotic arm to approach the garment, as we can notice the whole trajectory leads to a decrease of the z-coordinate and an increase of the x-coordinate.

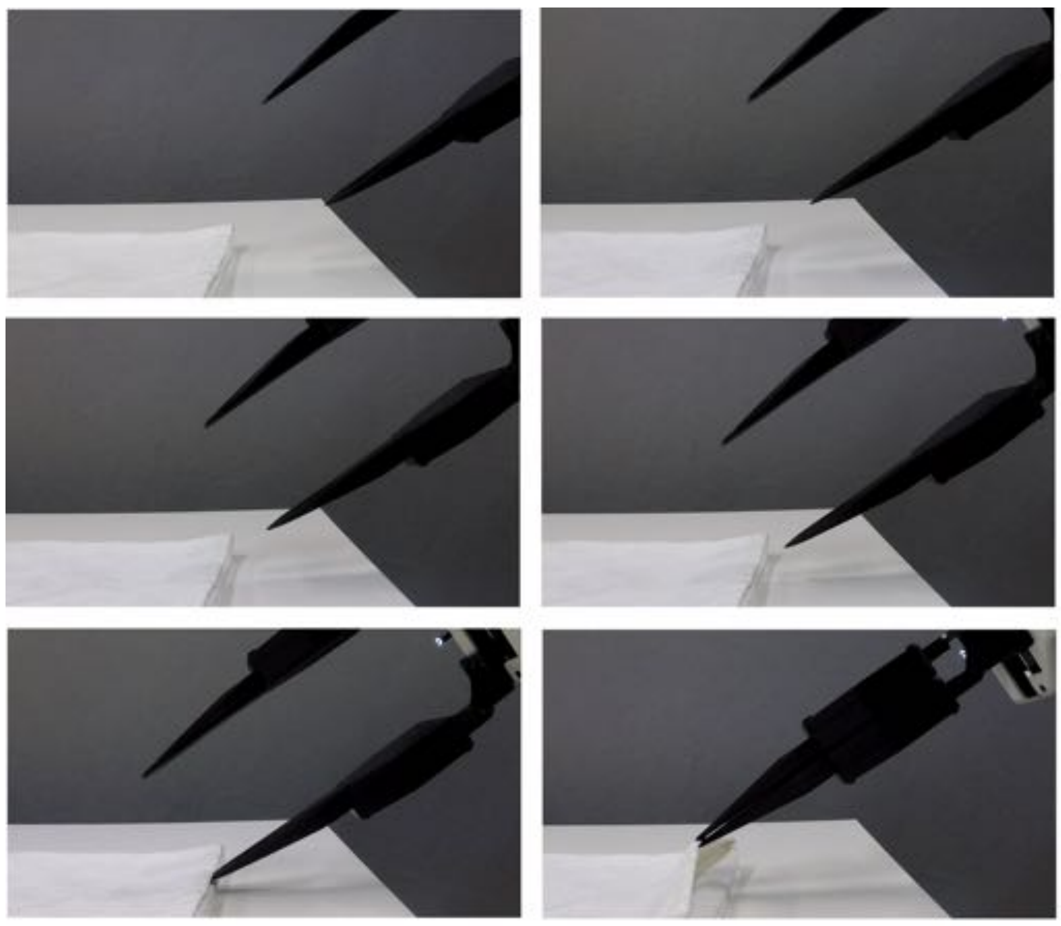

Figure 82: Typical movement of the end-effector towards the garment

5.4.1 Stability to changes of the garment

We are interested in a procedure that is as more general as possible, so we carried out tests with garments of different colors, textures and thicknesses to evaluate the versatility of the proposed procedure. The garments used for the experiments are shown in Figure 83 (in particular they are seen from the camera point of view at the beginning of the motion).

In every case the vision has correctly traced the line of interest, increasing the coordinate x or decreasing the coordinate z depending on the position of this line with respect to the edge of the end-effector.

Figure 84 shows the images elaborated from the vision part of the algorithm. On the left part of every figure there is the picture elaborated with the Sobel filter where has been highlighted in blue the threshold beyond which the lines are filtered, in light green the line corresponding to the tip of the end-effector and in darker green the line of interest. Instead on the right part there is the picture acquired from the camera with the Canny edges and in green the line corresponding to the tip of the gripper.

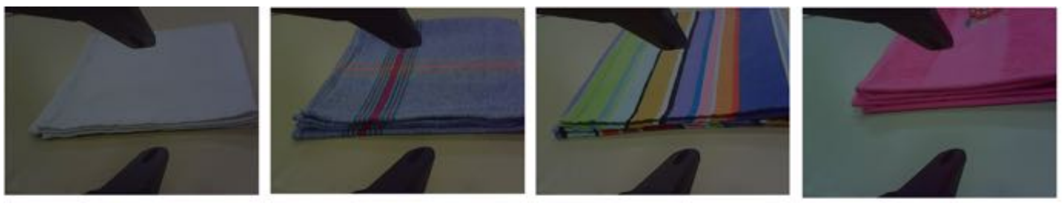

Figure 83: Types of garments used for the experiments

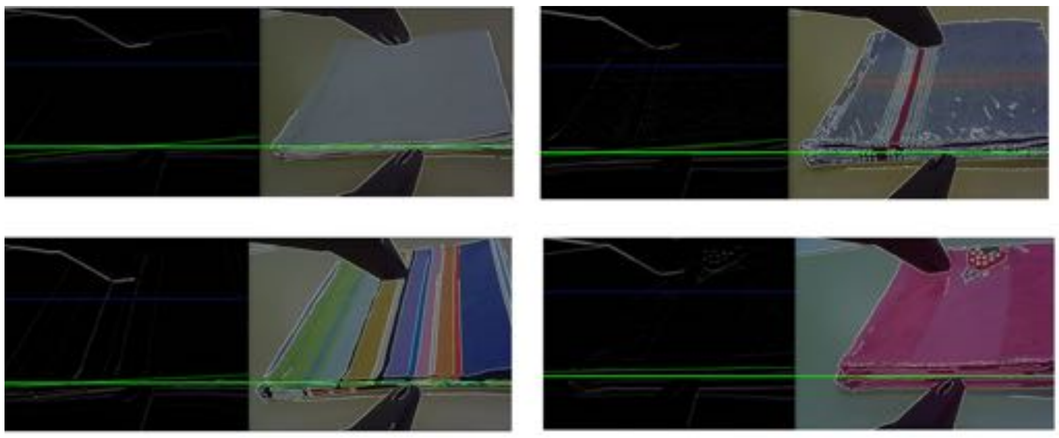

Figure 84: Vision results with different garments

If we look at the right part of the previous figures we can see that while in the case of white garment all the lines that do not belong to the edges of the garment are filtered out, in the other cases the Canny edge detector also detects other lines, namely:

- in the case of the textured garment in the image there is a lot of noise due to the diagonal segments of the texture and the vertical lines drawn on the surface of the garment,
- in the case of the striped garment all the vertical lines drawn on the cloth are detected,
- in the case of the pink garment case there is some noise near the border of the cloth and the edges of the strawberry drawn on the it are detected as lines.

Nevertheless with all types of garment the procedure is successful, because as we can see from the left side of the previous images, all the noise and most of the vertical lines are filtered by the Sobel filter and the Hough transform.

Three consecutive tests have been performed for every case and we obtained 3 successes out of 3 tests in each case.

We can therefore conclude that the procedure works correctly with different types of garment indifferently. This is positive because it means that the procedure is very general from this point of view.

The only case in which this method does not work is in presence of a garment with stripes parallel to the layers, because the stripes can be mistakenly identified as layers.

In the light of what we have just observed, without loss of generality in the following we perform the other tests only with the white garment.

5.4.2 Stability to changes of background surface

Then we tested the stability of the algorithm to changes of the surface on which the garment is placed. The purpose of these tests is to verify if changes of the surface, in particular of its colour and consequently of the amount of reflected light, can influence the performance of the algorithm.

We performed tests with the wooden-like surface, the white plastic surface and the blue surface shown in Figure 85.

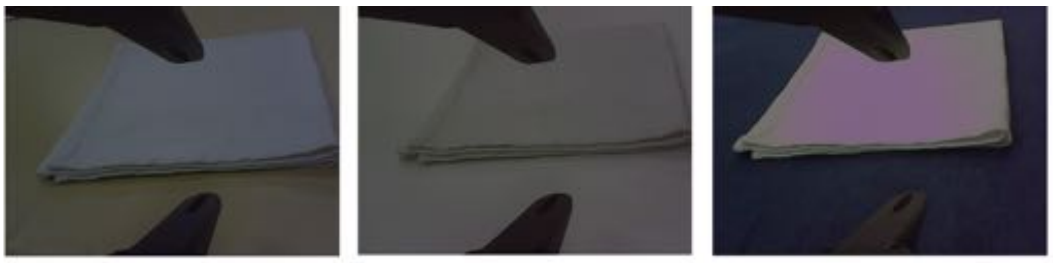

Figure 85: Initial scene perceived by the camera with different surfaces

As we can see from Figure 86, that contains images elaborated from the vision part of the algorithm, the edges recovered by vision in the three cases are almost the same.

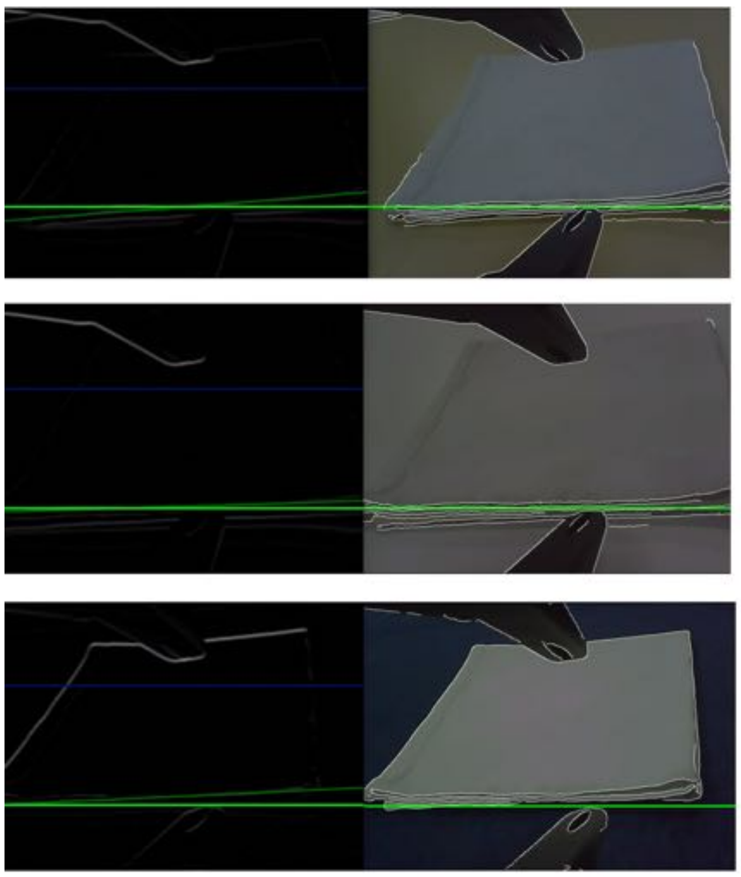

Figure 86: Vision results with different surfaces

For example in the right part of the first image we note that with the wooden-like surface the Canny edge detector identifies some lateral edges of the cloth, that with the white surface are not found (see the second image). But these edges are then discarded by the Sobel filter, indeed the left part of the first two images is almost equal. The last image shows the case with the blue surface, in this case we detect all the edges belonging to the perimeter of the garment. This is consistent because there is much more contrast between the cloth and surface than in the previous two cases.

However this does not compromise the result of the experiment because the upper edge, parallel to the layers, is filtered out with the threshold *end_range* and the diagonal one is not considered.

For every case we performed 3 tests and we obtained 3 successes because vision was able to correctly trace the line of interest, increasing the coordinate x or decreasing the coordinate z depending on the position of this line with respect to the edge of the end-effector.

We can therefore conclude that the procedure works correctly with different colors of the surface indifferently. This is positive because it means that the procedure is very general also from this point of view. In the light of what we have observed in this section, without loss of generality in the following we perform the other tests only with the white garment placed on the wooden-like raw surface of the table we have been using as standing support.

5.4.3 Stability to changes of the number of layers

After that we tested the stability of the algorithm to changes of the number of visible layers of the garment. We performed some tests with a single layer, 2 and 4 layers visible, Figure 87 shows the three initial configurations.

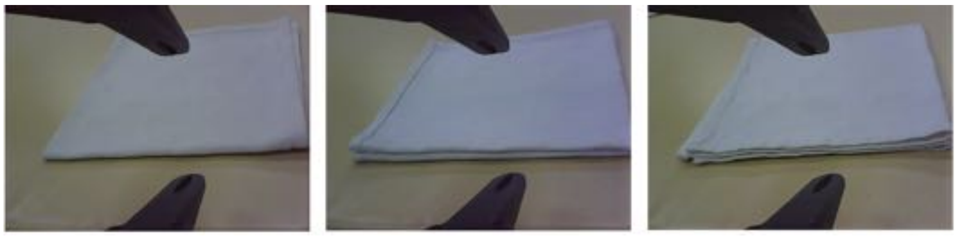

Figure 87: Initial scene with different number of layers

The effect of the vision is almost the same in the three cases, indeed as we can see from Figure 88, the result of the Canny edge detection with the Sobel filtering leads to the images on the left part, that differ only for the number of layers in the bottom.

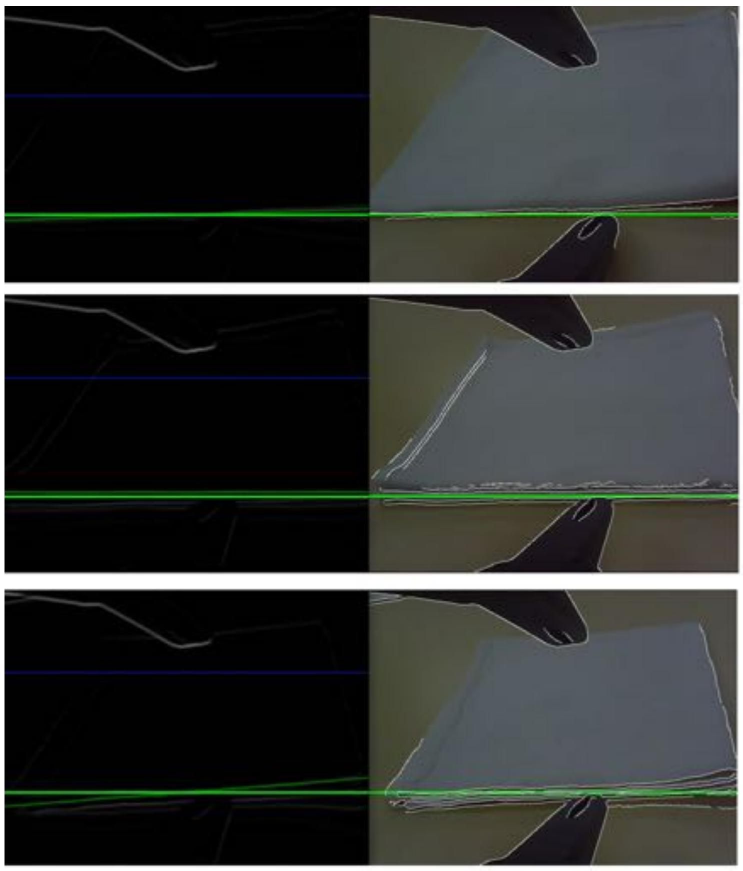

Figure 88: Vision result varying the number of visible layers

The average time required to perform the movement is almost the same in all cases, in fact in the case of 1 level the tests required an average time of 40 seconds, in the case of 2 layers an average time of 43 seconds and in the case of 4 levels an average time of 42 seconds. We performed 3 tests and in each case we obtained 3 successes, so we can affirm that the procedure is not affected by the number of layers.

Therefore, in the following without loss of generality we perform tests with 4 layers visible. This choice is due to the fact that this case is the one that requires more precision, because a layer is very thin, so an error of a couple of millimetres on the height could compromise the end of the task.

5.4.4 Stability to changes of initial position

Afterwards we assessed whether the success of the procedure could depend on the initial position of the garment in space. Therefore we carried out 3 consecutive tests with the cloth in the following positions:

- a central position, with the garment at 9cm from the border of the surface and the tip of the end-effector at 7cm from the garment,
- then we moved the garment up of 2cm (further away from the gripper),
- then up of other 2cm (so at 13cm from the border of the surface),
- after that from the initial position we moved the garment down of 2cm (so closer to the end-effector),
- down of other 2cm (at 5cm from the border of the surface).

Figure 89 schematizes the positions where we moved the cloth.

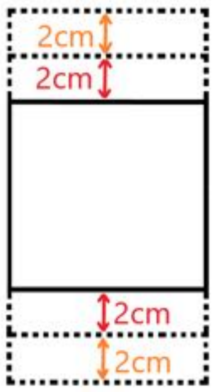

Figure 89: Scheme of the different initial positions that have been tested

Central position

With the garment in the central position each of the tests led to success. Figure 90 shows on the left the initial scene perceived by the camera, while on the right a top view of the position of the cloth and of the gripper in space.

We can notice that the cloth appears central in the image seen from the camera, but from the top view we can see that it is almost entirely to the left of the end-effector.

This is due to the fact that the camera is no longer integrated in the center of the end-effector, but it is fixed on the right side of the gripper, pointing to the tip of the end-effector.

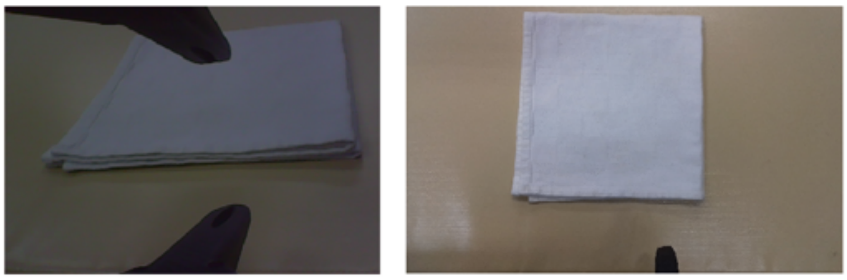

Figure 90: Central position of the cloth, camera and top view

The typical messages sent to the Whole Body Controller and the variation of the position of the arm-tool link during a motion are reported in Figure 91. In particular we can notice that the messages sent to the WBC have the expected step pattern, where we either increase the x coordinate or decrease the height, without ever changing them together.

Even if the shape of the trajectory of the arm-tool link is similar to the sequence of messages sent to the WBC, we observe that there is a large error between the actual x-coordinate and the one sent to the WBC.
As explained before, this is due to the fact that the gains of the Whole Body Controller are too low, so, if we send increments on the x-coordinates of a few millimetres, as would be logical considering that we have to make small movements, the complete movement would take too much time.

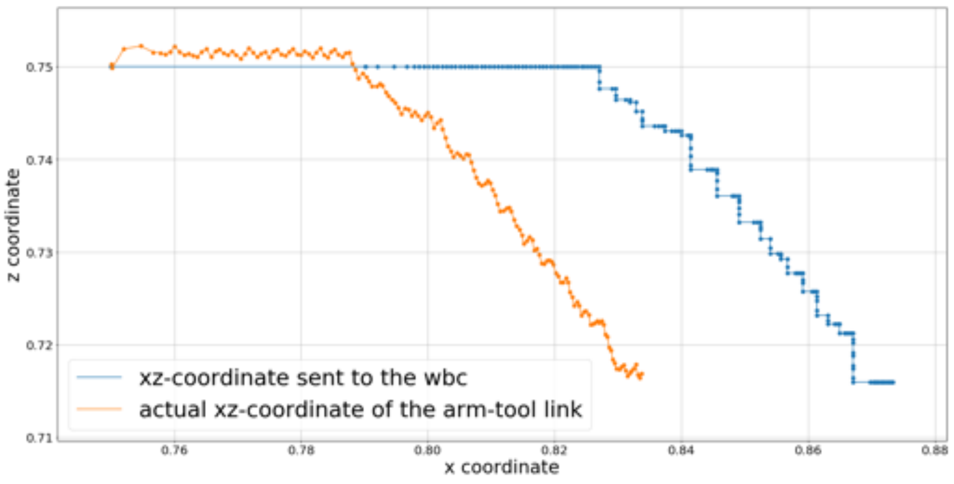

Figure 91: Typical xz variation during the motion

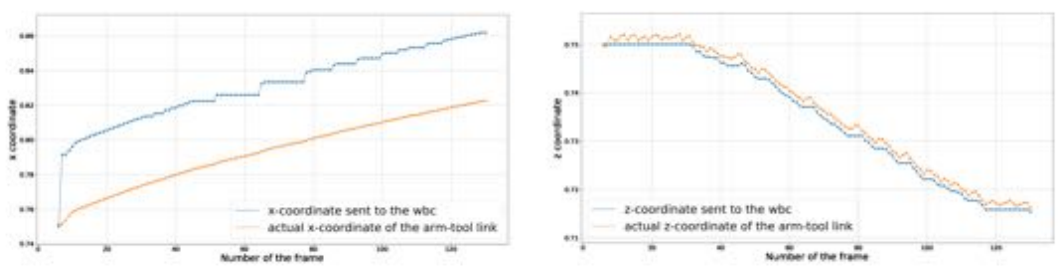

Figure 92: Typical x and z variation during the motion

For example we tried to send increments $\delta x = 0.006m$ and the total movement (which is of 10-20 cm along x) took 5 minutes, an unacceptable time considering the distances involved.

To avoid changing the gains of the WBC, since it is a delicate operation and would compromise the collaborative characteristics of the robot, we preferred to send very large x increments. In particular, we performed tests sending increments $\delta x = 4cm$, $5cm$ and $6cm$.

Obviously the arm-tool link is not able to cover such a big distance before receiving the next message, but we exploit these increments to indirectly act on the speed of the motion. In particular, with increments $\delta x = 4cm$, the robot is able to complete the movement in 40 seconds. This is the reason why we can observe what it seems a big error on the x-coordinate in Figure 91.

For movements along the z-direction the robot mainly uses the torso, so it performs a linear movement using a prismatic joint. Since to modify the height the WBC does not have to involve the entire arm, the robot is able to quickly perform the commanded movement without having to exploit the same technique used for the x-coordinate.

In Figure 92 is reported the variation of x and z as the frames change. In particular from these graphs is evident that the z-coordinate is following the reference with a small error and the x-coordinate is varying similarly to the reference, even if there is this big error.

If we do not correct the controller as explained in section 5.2.4, the x and z coordinates would have a behaviour like the one shown in Figure 93. From this figure we can notice that both coordinates would have a very oscillatory trend. In particular, from the graph on the left we can see that every decrease of the z coordinate would coincide with a peak down in the message sent to the WBC and with a decrease of the actual x coordinate of the arm-tool link.

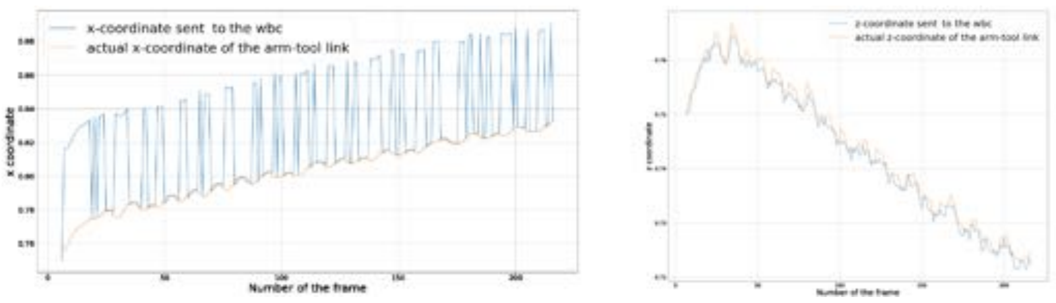

Figure 93: Typical x and z variation during the motion without correcting the controller

This means that at each decrease of the height of the arm-tool link we would have a movement of the gripper further from the garment in the x-direction. Similarly, from the graph on the right we can see that the height of the gripper would be increased at each increase of the x-coordinate. Comparing these graphs with those reported in 92 we can conclude that the adopted strategy is very effective to reduce this type of unwanted fluctuations.

Variations of the position

We are then interested in how the trajectories vary with respect to the initial position, to understand how much general is this procedure.

Figure 94 shows the graph of the x and z coordinates with the garment in the central position, moved up and moved down.

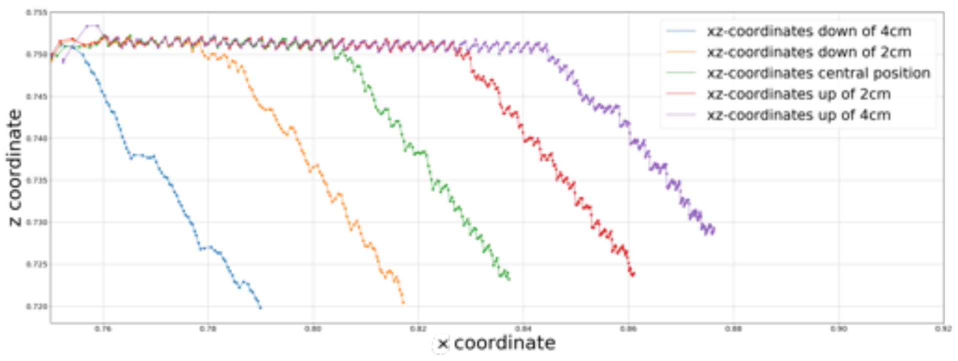

Figure 94: xz trajectories with the different initial positions

We can note that the x coordinates, whose variation with respect to the number of frames is shown in Figure 95, has a reasonable behaviour.

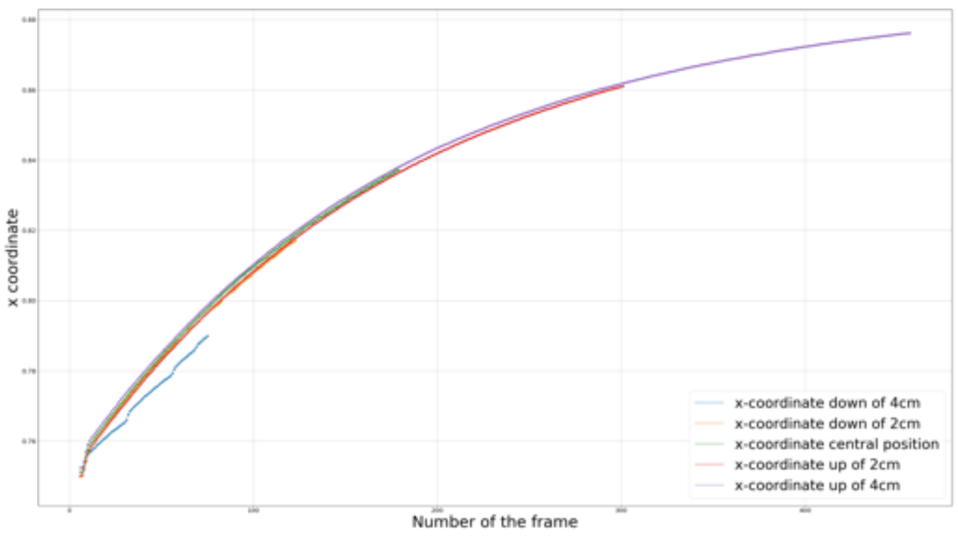

Figure 95: x trajectories with the different initial positions

Indeed the trajectories end at 0.792m, 0.818m, 0.838m, 0.861m and 0.878m from the base, consistently with the fact that each time we moved the garment further of 2cm from the previous position.

The z coordinate, whose variation depending on the number of frames is shown in Figure 96, also varies coherently.

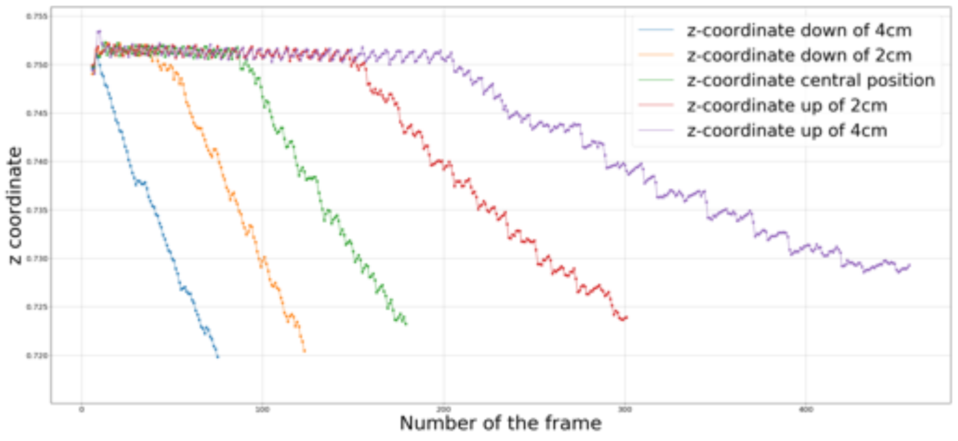

Figure 96: z trajectories with the different initial positions

In fact, when the garment is closer to the end-effector, the height starts to decrease sooner and decreases faster, while when the garment is further the trajectory decreases less steeply.

Moreover the final height is consistent in the first positions, because it varies between 0.720m and 0.724m, while when the garment is in the furthest position from the robot, it is 4mm greater. This may be due to the fact that the arm is close to its maximum extension, so the controller tries to avoid to reach the singular configuration that coincides with the alignment of all the joints, causing bigger oscillations of the arm-tool link.

Since in every case the three tests ended with successful grasps, we can conclude that the initial position of the cloth is irrelevant for the result of the experiment, this is positive because we do not have loss of generality.

Variations of the orientation

One of the initial assumptions of this procedure is to have garments placed with the layers perpendicular to the x-direction of the camera. We were interested in evaluating the strictness of this assumption, so we performed a couple of tests with the garment not perfectly perpendicular, but rotated of about 30° with respect to the perpendicular.

Both the tests ended successfully, so we can qualitatively affirm that the procedure works also with garments rotated by less than $+/-30°$ from the perpendicular. If we want to make this method work even with garments more rotated we would need to develop a new technique to choose the layer to be grasped, for example taking into account the orientation of the line of interest.

Figure 97 shows on the left the initial configuration of the tests, in the center the result of Sobel filtering (with the line of interest depicted in dark green) and on the right the picture acquired from the camera with the Canny edges.

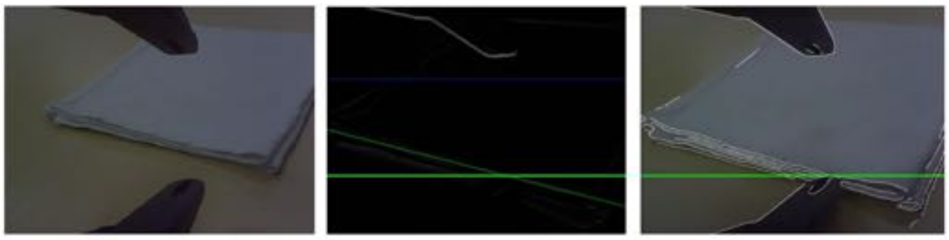

Figure 97: Initial configuration of the experiments with the garment rotated of about 30° with respect to the perpendicular, result of Sobel filtering and image from the camera with the Canny edges

5.4.5 Stability to repetitions

Another aspect that we analyzed is the stability to N subsequent repetitions to test the repeatability of the procedure.

We performed $N = 10$ tests with different increments of the x coordinate, starting from $\delta x = 4cm$, then with $\delta x = 5cm$ and finally $\delta x = 6cm$.

Figures 98 and 99 show the variations of x and z coordinates in the 10 tests with $\delta x = 4cm$.

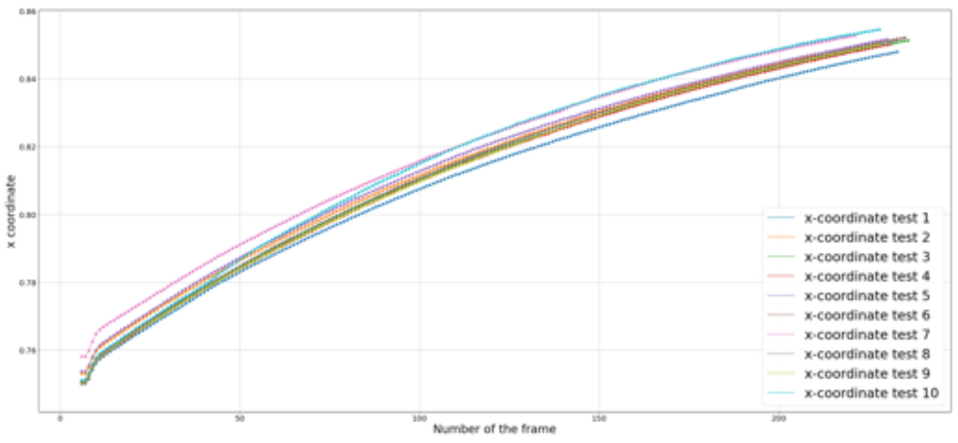

Figure 98: Variation of the x-coordinate in the 10 tests, $\delta x = 4cm$

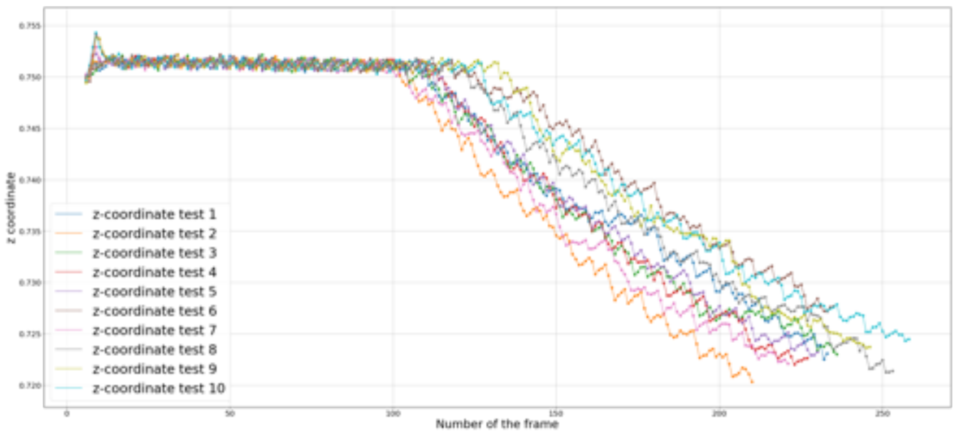

Figure 99: Variation of the z-coordinate in the 10 tests, $\delta x = 4cm$

This case all the tests ended with a success, so even if the trajectories performed by the arm change slightly at each trial, we have a percentage of successes over tests of 100%.

Increasing δx to $5cm$, the success rate decreased to 90%, because we obtained 9 successes out of 10 tests.

Indeed, as we can see from Figures 100 and 101, that show the changes in the x and z coordinates as the frames vary, the tenth test failed. In this case the gripper missed the last level of the garment, ending higher and further than the right position, this can be seen from the fact that both the coordinates at the end of the movement are significantly higher than in the other tests.

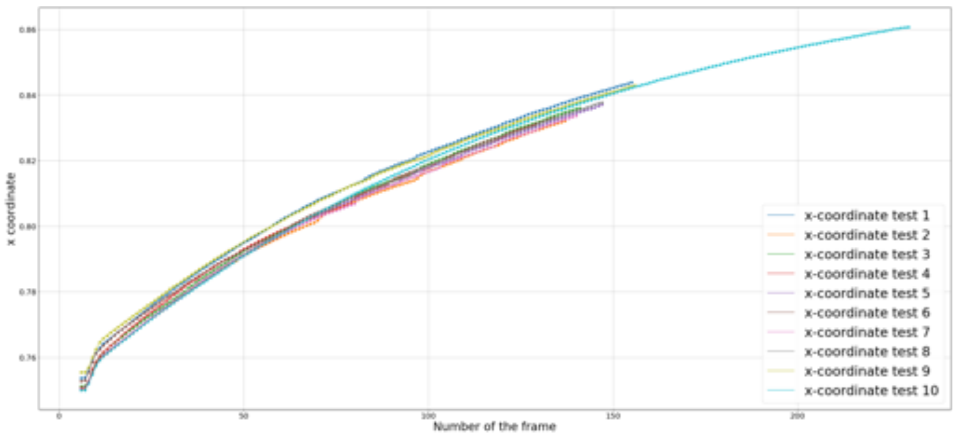

Figure 100: Variation of the x-coordinate in the 10 tests, $\delta x = 5cm$

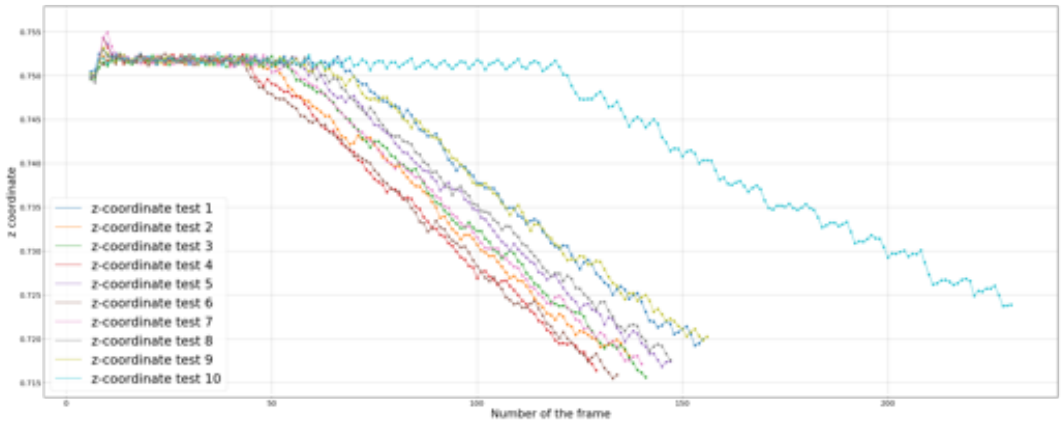

Figure 101: Variation of the z-coordinate in the 10 tests, $\delta x = 5cm$

Finally, taking $\delta x = 6cm$, the success rate was 70%, in fact, as can be seen from Figures 102 and 103, the 1st, the 6th and the 10th tests failed.

In particular during the 10th test the gripper missed the layer ending too high and too far from the objective, while in the other two cases the height was decreased too quickly and the gripper grabbed not only the first but also the second layer of the garment.

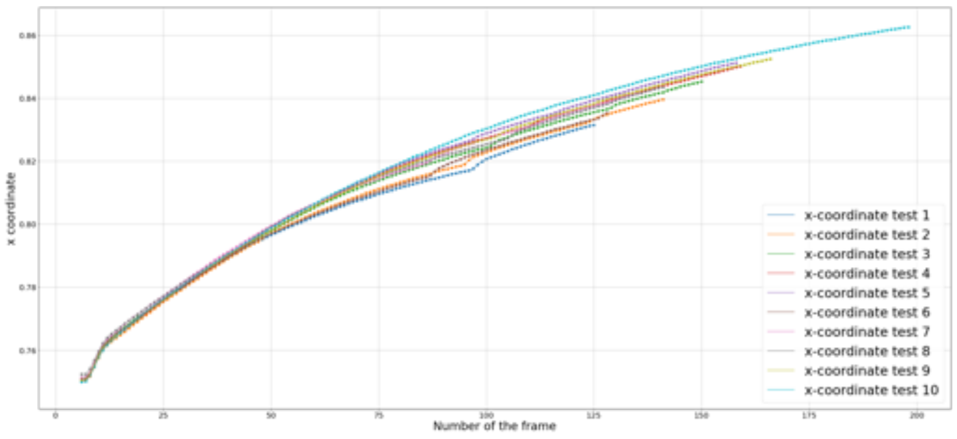

Figure 102: Variation of the x-coordinate in the 10 tests, $\delta x = 6cm$

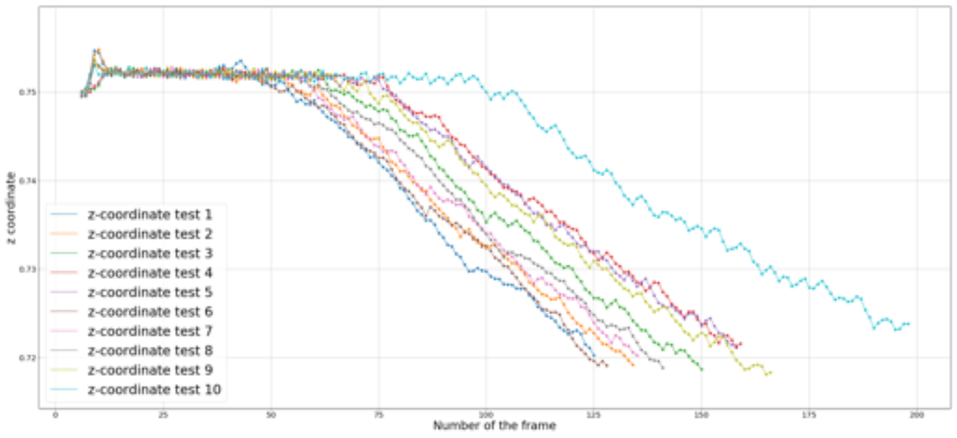

Figure 103: Variation of the z-coordinate in the 10 tests, $\delta x = 6cm$

We can therefore conclude that if we do not take too high x-increments this procedure is stable even in consecutive repetitions.

Stability to repetitions in worse lighting conditions

We performed 10 consecutive tests also with the source of strong light, with increments $\delta x = 4cm$, to check if the percentage of successes varied.

As we can see from the Figures 104 and 105, which report the variations of x and x as the frames vary, we got 9 successes out of 10 tests. Indeed in test 9 the gripper missed the highest layer, ending too high and too far.

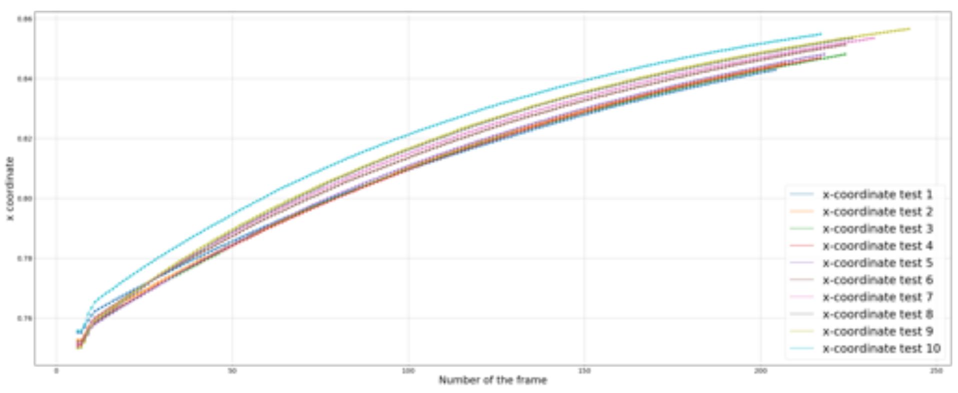

Figure 104: Variation of the x-coordinate in 10 tests, $\delta x = 4cm$, strong light

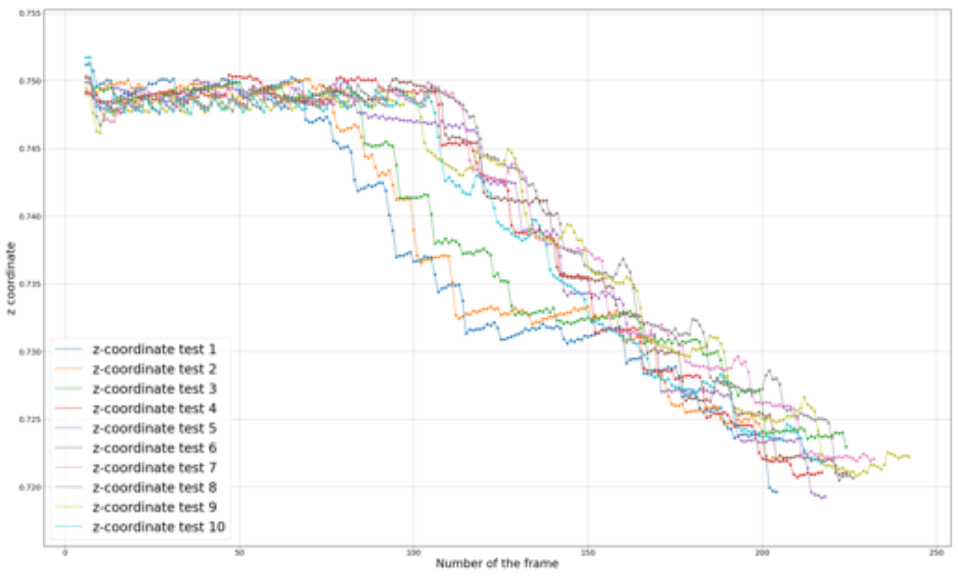

Figure 105: Variation of the z-coordinate in 10 tests, $\delta x = 4cm$, strong light

We can also note that, although all the other tests ended with a successful grasp, the variations of the coordinates have been very different: in some cases the z-coordinate started to decrease very soon and then stabilized (as in test 1), in other cases, the height started decreasing later (as in tests 7 and 8).

This variability is due to the fact that in these light conditions in the first part of the movement the vision is very unstable, so different behaviours are due to the different position where the line of interest is detected in the first frames.

We can conclude that in this lighting conditions the performances are worse than in the other case, because the percentage of successes is 90% and the trajectories vary more, but the result is still quite good.

5.5 Analysis of the performances of the approach

Then we wanted to assess the overall performance of the procedure, in terms of timing. We first analyzed the time required at each cycle (which means for each frame) to perform the following operations:

- detect the edges in the image, with Canny edge detector,
- filtering the image with Sobel filter,
- extract the lines in the image with Hough transform,
- search the line of interest among the ones detected,
- update the line in memory,
- compute the increments and send them to the WBC.

After that we evaluated the average times required for the whole movement of the robotic arm from the initial position until reaching the layer, just before the closing of the gripper.

5.5.1 Average times required by the principal parts of the code

Figures 106 and 107 show the times required to perform the above operations in the case of tests with standard laboratory illumination (so 554-560 lux) and with the strong light source (825-832 lux).

From these figures we can immediately notice that the operation that requires an higher time than the others is the extraction of the lines with Hough transform, which takes between 0.03 and 0.07 seconds for each image.

This is consistent because this operation works on all the edge pixels of an image that has 640x480 pixels. To speed up this operation we should downsize the image or change the parameters.

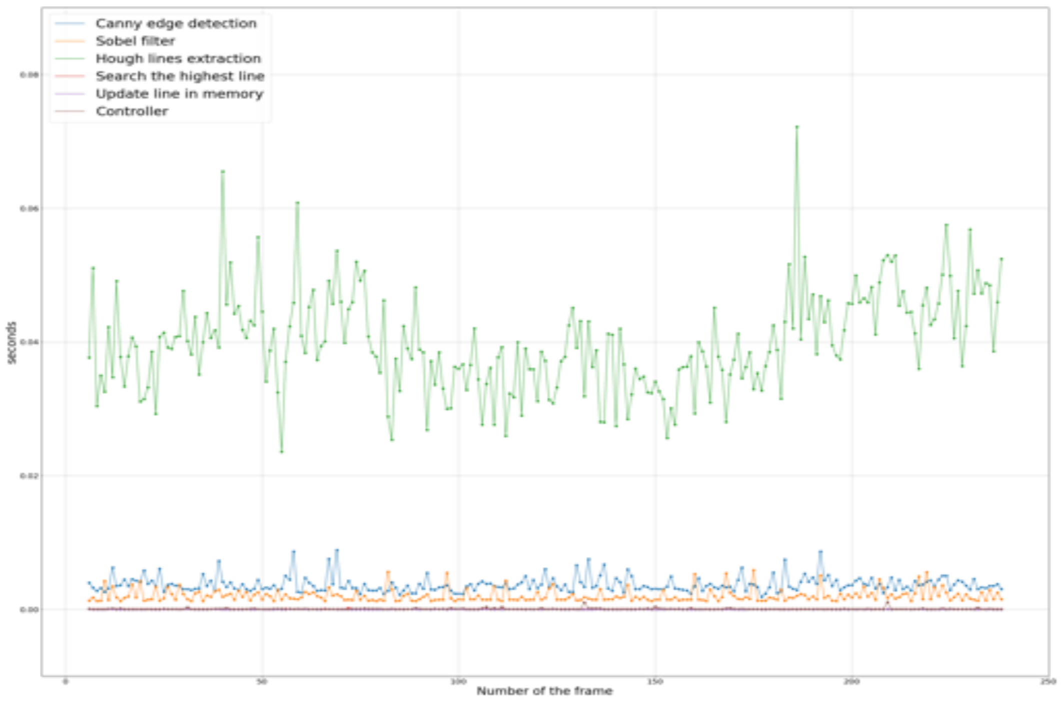

Figure 106: Times required for principal operations, normal lighting

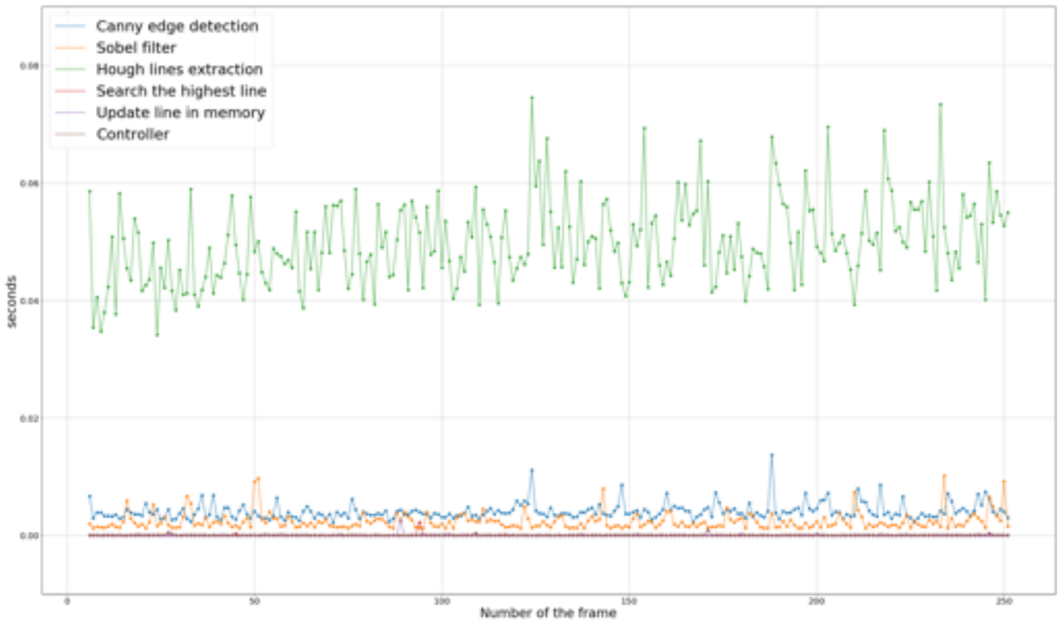

Figure 107: Times required for principal operations, strong lighting

The other more time-consuming operations are Sobel filtering and Canny edge detection that require 0.001-0.005s and 0.001-0.01s respectively.

Finally the operations that require less time are the research of the highest line among the ones detected, and the update of the line in memory, that require times of the order of 10^{-5} and $10^{-6}s$ respectively.

Figures 108, 109 and 110 show the comparison of the times requested for each operation in the case of normal lighting and in the case of strong lighting.

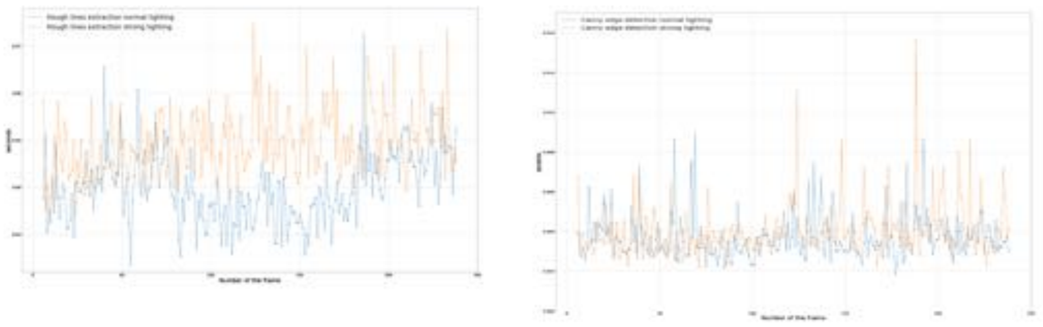

Figure 108: Times required by Hough transform and Canny edge detector in the two different lighting conditions

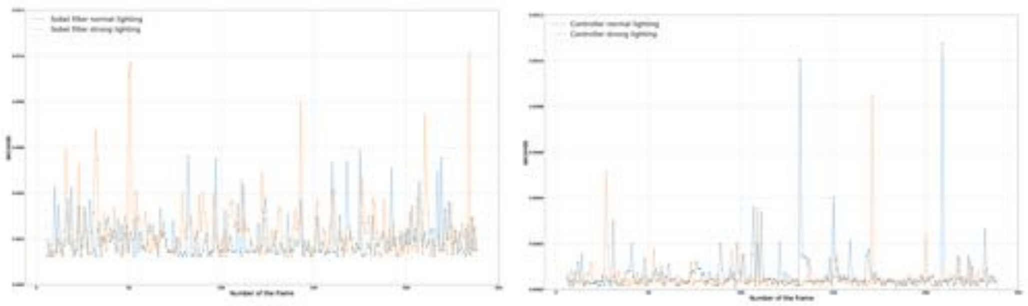

Figure 109: Times required by Sobel and to compute the increments in the two different lighting conditions

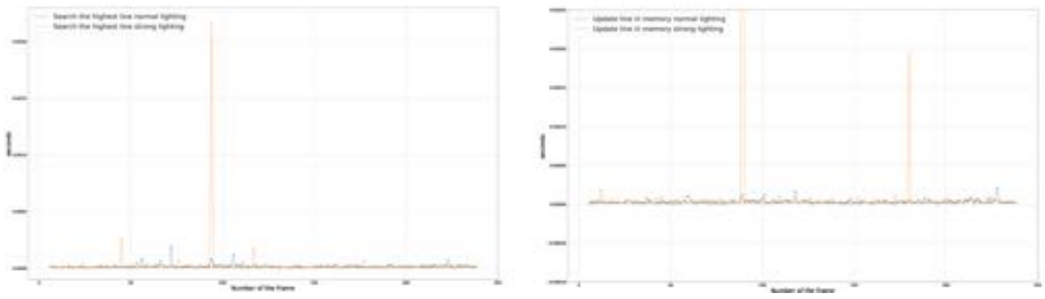

Figure 110: Times required to compute the highest line and to update the one in memory in the two different lighting conditions

We can notice that these times are approximately the same, except for the

Hough transform that in the case of normal lighting requires slightly less time and at half of the movement decrease a bit. As regards Sobel filtering, we can see that it requires less time with the normal lighting.

5.5.2 Average times to complete the task

Then we inspected how the velocity of the whole movement, from the initial position until when the garment is reached, varies depending on the initial position of the garment and on the increment δx.

Table 2 reports the average times required in the different cases.

Position of the garment	**δx[cm]**	**Time [s]**	**Number of frames**
Down of 4cm	4	23	75
Down of 2cm	4	29	120
Central	4	42	190
Up of 2cm	4	51	300
Up of 4cm	4	110	440
Central	5	35	140
Central	6	40	145

Table 2: Average times needed to perform the whole movement

6 Budget

In this section we estimate the total cost of this project in terms of hardware and software resources and working hours of people involved in the work.

6.1 Hardware cost

The main hardware resources involved in this project are a TIAGo robot, a 3D printed gripper and a computer of the laboratory.
We estimate that the robot has been used for approximately 4 weeks more or less 6 hours a day, 5 days a week, for a total of 120 hours.
Instead the computers have been used for approximately 20 weeks, 5 days a week for more or less 7 hours a day for a total of approximately 700 hours.

Both the robot and the computer have a life expectancy of 5 years, so, considering that the robot costs 50000€ and a computer about 2000€, their cost per hour is more or less 4.20€/h and 0.20€/h.
Therefore their cost related to this project is 504€ and 140€ respectively. The cost of printing in 3D the gripper is negligible compared to the other costs.

Regarding the electric consumption, a computer requires an average power of 200W, so we have consumed approximately 140kW of electricity. We estimate that the robot needs an average power of 400W, so the electricity consumption is 48kW. Considering that in 2021 the electricity prices in Spain are about 0.17€/kWh, we evaluate that the total energy cost related to the project of 32€.

To sum up we estimate that the total cost of having used these resources during the months that this project has last is of 676€.
The prices and characteristics of these resources are summarized in the following table.

Hardware resource	**Life expectancy**	**Cost (€)**	**Hours of use**	**Energy consumption**
TIAGo robot	5 years	50000	120	400W
Gripper	1 year	40	120	0W
Computer	5 years	2000	700	200W

Table 3: Cost of hardware resources

6.2 Software cost

The software resources, that are Robot Operating System (ROS) and the C++ compiler are free, so the total cost of the software licenses is of 0€.

6.3 Human resources cost

Finally we have to take into account the salary of people involved in the project. The project has been developed in 5 months, the supervisor of the project and other members of the laboratory have spent approximately 5 hours a week for a total of 100 hours with an average cost of 40€/h.
The developer has spent about 8 hours a day, 7 days a week, for a total of 1120 hours in 20 weeks, with an average cost of 30€/h.

The following table summarizes the total cost of human resources involved in this work.

Role	**Price per hour (€)**	**Total hours**	**Salary**
Supervisors	40	100	4000
Engineer	30	1120	33600

Table 4: Cost of human resources

The total cost of the project finally is of 38276€.

7 Environmental and social impact

Finally, we want to reflect briefly on the environmental and social impact of this work.

Environmental impact

For this project a gripper has been printed in 3D, however it was built using a polylactide plastic (PLA) that is recyclable, so from this point of view the environmental impact is almost null.
The software part of this work does not have a direct impact on the environment, however we have to take into account the energy consumption of the robot and the computer. At the time of writing this document, the website *https://www.electricitymap.org/zone/ES* reports a production of CO_2 per kWh in Spain in the last 24 hours that varies between $70gr/kWh$ and $90gr/kWh$. Considering an average of $80gr/kWh$, if we consumed about 188kW, the amount of CO_2 produced during the development of this project is of 15kg.

Social impact

Social and assistive robotics has experienced a great progress in the last years. This is due to the fact that the world's population is aging and the number of care personnel is decreasing, causing quite high healthcare costs. Introduce into our daily lives cooperative robots will probably increase the well-being and the independence of people with disabilities or seniors, helping them to complete some simple tasks without the necessity of attendants.
In particular the CLOTHILDE project is developing a theory of cloth manipulation that will probably be very supportive in helping disable or old people to autonomously grasp clothes and dress up. In future, with robots capable of completing even more complex tasks, some jobs positions may disappear, but at the same time new types of jobs, that require different skills, will arise.

8 Conclusions

This work has studied the feasibility of using a subjective camera, mounted on the robot arm end-effector, to perform delicate manipulations on a garment. In particular, we have concentrated on grasping the top layer of a folded garment. We have shown that, when the camera position is appropriate, the obtained images contain some relevant details that are very valuable to close the control loop. We have used a robot control mode, the Whole Body Controller, that has benefits because accepts continuous updates on the goal and assures safety with human contact, but at the price of low precision.

The evaluation that we performed show the validity of the proposed method when we deal with folded garments with an unknown number of layers. This method is not influenced by the texture of the garment, neither by its thickness nor by the color of the surface on which it is placed. This procedure works also in different non-extreme lighting conditions. If the procedure has to be applied in particularly strong or weak light conditions, more complex segmentation techniques should be used to process the image. In our case we preferred these rather simple techniques to speed up the image processing phase.

In terms of the average time required for the entire movement the performance is reasonable. Future improvements may include the tunning of the gains of the Whole Body Controller in order to avoid the necessity of sending high increments on the x-coordinate to increase the motion speed.

We have limited the scope to grasping of the middle section of the top layer, but the method can be broaden to grasp other parts, like one corner, or other layer combinations, like the two top layers. This would require to enlarge the perception algorithms with new functionalities. Nevertheless, the approach including the subjective camera and the closed-loop control strategy would be valid.

A limitation of this strategy is that the stopping criteria does not work with black or too dark garments, if we want to extend the validity of the procedure also to these cases it will be necessary to elaborate a new criterion to end the movement of the robot. For example, we may segment the image according to the different textures of the objects, or we may use sensors on the finger tips, or the measurements of the force sensor placed on the arm-tool link of the robot to understand when the garment has been touched.

References

[1] Da Vinci surgery, "About da Vinci Systems. Surgical robotics for minimally invasive surgery" Last access: 12/11/2020. [Online]. Available: https://www.davincisurgery.com/da-vinci-systems/about-da-vinci-systems

[2] Ecorobotix, "Avo weeding robot" Last access: 12/11/2020. [Online]. Available: https://www.ecorobotix.com/en/avo-autonomous-robot-weeder/

[3] German Research Center for Artificial Intelligence GmbH, Robotics Innovation Center, "Robotics fields of application" Last access: 12/11/2020. [Online]. Available: https://robotik.dfki-bremen.de/en/research/fields-of-application.html

[4] C. Torras, "Service robots for citizens of the future," European Review, vol. 24, no. 1, pp. 17–30, 2016.

[5] Julia Borràs, Guillem Alenyà, Carme Torras, "A Grasping-centered Analysis for Cloth Manipulation", IEEE Transactions on Robotics, vol. 36, no. 3, pp. 924-936, 2020.

[6] Project website "Clothilde, Cloth manipulation learning from demonstrations", Last access: 13/11/2020. [Online]. Available: https://clothilde.iri.upc.edu/

[7] Kyoko Hamajima, Masayoshi Kakikura, "Planning strategy for task of unfolding clothes", Robotics and Autonomous Systems, Volume 32, Issues 2–3, 2000, Pages 145-152.

[8] Dimitra Triantafyllou, Ioannis Mariolis, Andreas Kargakos, Sotiris Malassiotis, Nikos Aspragathos, "A geometric approach to robotic unfolding of garments", Robotics and Autonomous Systems, Volume 75, Part B, 2016, Pages 233-243.

[9] Jeremy Maitin-Shepard, Marco Cusumano-Towner, Jinna Lei, Pieter Abbeel, "Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding", 2010 IEEE International Conference on Robotics and Automation, 2010.

[10] P. Jiménez, "Visual grasp point localization, classification and state recognition in robotic manipulation of cloth: An overview", Robotics and Autonomous Systems, Volume 92, 2017, Pages 107-125.

[11] Radu Horaud, Fadi Dornaika, "Hand-eye Calibration". The International Journal of Robotics Research, SAGE Publications, 1995, 14 (3), pp.195–210.

[12] Irene Garcia-Camacho, “Robotic manipulation skills for picking and unfolding garments”, Last access: 16/11/2020. [Online]. Available: https://www.iri.upc.edu/master_thesis/show/140

[13] Espinosa Muñoz, Miller Stiven, “Mobile manipulation with the TIAGo robot: perception and task manager”, Last access: 16/11/2020. [Online]. Available: https://upcommons.upc.edu/handle/2117/166932

[14] April robotics laboratory, “AprilTag”, Last access: 17/11/2020. [Online]. Available: https://april.eecs.umich.edu/software/apriltag

[15] Sung-Hoon Im, Gyeongmin Choe, Hae-Gon Jeon, In So Kweon, “Depth from accidental motion using geometry prior”, Last access: 18/11/2020. [Online]. Available: https://sunghoonim.github.io/assets/paper/ICIP15_Depth.pdf

[16] M. A. Arteaga, M. Bueno-Lopez and A. Espinosa, “A simple approach for 2D visual servoing”, 2009 IEEE Control Applications, (CCA) & Intelligent Control, (ISIC), St. Petersburg, 2009, pp. 1557-1562.

[17] Ezio Malis, François Chaumette, Sylvie Boudet, “2 1/2 D Visual Servoing”, IEEE Transactions on Robotics and Automation, Institute of Electrical and Electronics Engineers (IEEE), 1999, 15 (2), pp.238-250.

[18] Xavi Gratal, Javier Romero, Jeannette Bohg, Danica Kragic, “Visual servoing on unknown objects, Mechatronics”, Volume 22, Issue 4, 2012, Pages 423-435.

[19] Don Agravante, Giovanni Claudio, Fabien Spindler, François Chaumette, “Visual servoing in an optimization framework for the whole-body control of humanoid robots”, IEEE Robotics and Automation Letters, IEEE 2017, 2 (2), pp.608-615.

[20] A. Mitjans, M. Maceira and G. Alenyà. “Corner detection of deformable fabric using deep learning”, Technical Report IRI-TR-20-01, Institut de Robòtica i Informàtica Industrial, CSIC-UPC, 2020.

[21] Jianing Qian, Thomas Weng, Luxin Zhang, Brian Okorn, David Held, “Cloth Region Segmentation for Robust Grasp Selection”, Last access: 30/01/2021. [Online]. Available: https://sites.google.com/view/cloth-segmentation.

[22] Fraś J., Maciaś M., Czubaczyński F., Sałek P., and Główka J., “Soft flexible gripper design, characterization and application”, Last access: 21/11/2020. [Online]. Available: https://jfras.github.io/jfras/papers/scit2016.pdf

[23] Brian Heater, Lora Kolodny, “How marine biology inspired Soft Robotics’ industrial grippers”, Last access: 21/11/2020. [Online]. Available: https://techcrunch.com/2017/04/01/soft-robotics-grippers/

[24] "Soft, flexible gripper uses Gekco-inspired adhesives", Last access: 21/11/2020. [Online]. Available: https://www.slashgear.com/soft-flexible-gripper-uses-gekco-inspired-adhesives-11527027/

[25] S. Donaire, J. Borras, G. Alenyà, C. Torras, "A versatile gripper for cloth manipulation," 2020.

[26] "TIAGo technical specifications", Last access: 21/11/2020. [Online]. Available: https://pal-robotics.com/wp-content/uploads/2019/07/Datasheet_TIAGo_Complete.pdf

[27] "Overview of the CloPeMa robot", Last access: 21/11/2020. [Online]. Available: http://clopemaweb.felk.cvut.cz/the-robot/