Universidade de Aveiro
Departamento de Eletrónica,
Telecomunicações e Informática
2016

César Miguel
Rodrigues de Sousa

Controlo Visual de uma Cabeça Humanóide Usando Alvos Fixos

Visual Servoing of a Humanoid Head Using Fixed Targets

Universidade de Aveiro
2016

Departamento de Eletrónica,
Telecomunicações e Informática

César Miguel Rodrigues de Sousa

Controlo Visual de uma Cabeça Humanóide Usando Alvos Fixos

Visual Servoing of a Humanoid Head Using Fixed Targets

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Automação Industrial, realizada sob a orientação científica do Doutor Filipe Miguel Teixeira Pereira da Silva, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor Vítor Manuel Ferreira dos Santos, Professor Associado do Departamento de Engenharia Mecânica da Universidade de Aveiro.

o júri / the jury

presidente / president

Prof. Doutor Pedro Nicolau Faria da Fonseca
Professor Auxiliar do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

vogais / examiners committee

Prof. Doutor António Manuel Ferreira Mendes Lopes
Professor Auxiliar do Departamento de Engenharia Mecânica da Faculdade de Engenharia da Universidade do Porto

Prof. Doutor Vítor Manuel Ferreira dos Santos
Professor Associado do Departamento de Engenharia Mecânica da Universidade de Aveiro

agradecimentos /
acknowledgements

Quero expressar o meu profundo agradecimento aos meus orientadores, Professor Filipe Silva e Professor Vítor Santos, por me ajudarem a focar nos objetivos, pelo entusiasmo, confiança e os valiosos ensinamentos transmitidos ao longo do projeto.

Agradeço também a todos os meus colegas do laboratório de Automação e Robótica pelo companheirismo e boa disposição, especialmente ao Ricardo Pascoal, Jorge Almeida e João Torrão pelas ajudas e conselhos constantes ao longo do projeto que em muito contribuíram para a sua concretização.
Ao Eng. Festas pela ajuda na concretização do novo pescoço do humanoide e à empresa Motofil por disponibilizar o robô que permiu realizar a parte experimental da dissertação.

À ESSUA pela disponibilização do equipamento VICON, nomeadamente ao Professor António Amaro, reponsável pelo equipamento e ao Sr. Mário Rodrigues pelo acompanhamento e ajuda em todo o processo.

A todos os meus amigos que me acompanham (excecionais!), um especial agradecimento à Joana e à Ana pelo valioso feedback da escrita desta dissertação e ao Carlitos, Renato e Daniela pela ajuda na obtenção de informação nas áreas da fisiologia e neurociência.

À (Univer)cidade de Aveiro por tudo o que me deu.

Por último, mas de todo não menos importante, à minha família que está sempre presente, especialmente os meus pais pelo apoio e suporte incondicionais. Obrigado!

Palavras Chave

Controlo Visual, Visão por Computador, Cabeça Robótica, Alvo Fixo e Estimação de Movimento

Resumo

Este trabalho apresenta como tese que a visão pode ter um papel importante no equilíbrio e navegação de robôs humanóides tal como acontece nos seres humanos, em particular se se assumir a existência de características fixas no cenário envolvente. O Projeto Humanóide da Universidade de Aveiro (PHUA) é usado neste trabalho como base para desenvolver a proposição desta dissertação. Todos os componentes mecânicos do pescoço do PHUA foram reconstruídos e melhorados para assegurar uma infraestrutura fiável. Foram desenvolvidos algoritmos de processamento de imagem e seguimento visual para encontrar e seguir um alvo fixo, com o intuito de obter realimentação visual para o sistema de seguimento do pescoço. Desenvolveu-se também um algoritmo de controlo de seguimento para a cabeça do humanoide com o intuito de seguir um alvo baseado em realimentação visual. A informação da posição do pescoço pode ser integrada posteriormente com a rede sensorial do humanóide de forma a melhorar o equilíbrio do robô. Foram ainda calculadas e testadas as equações que estimar o movimento do robô, recorrendo aos ângulos da *pan and tilt unit* (pescoço) e sabendo a distância em cada instante da câmara ao alvo a seguir. O desenvolvimento do software foi baseado numa plataforma modular que permite a criação de vários modos de funcionamento independentes (ROS). Para simular os movimento do humanóide com a intenção de testar o sistema de seguimento desenvolvido, foi utilizado um robô industrial Fanuc. Os resultados dos testes demonstraram que os algoritmos de visão por computador tem um bom desempenho face ao contexto da aplicação. O controlo de seguimento baseado em velocidade, é o melhor para obter um sistema de seguimento visual para robôs humanóides simples e fiável.

Keywords Visual Control, Computer Vision, Robotic Head, Fixed target and Ego-motion

Abstract

Assuming the existence of fixed characteristics on the scene, this work addresses the thesis that vision may play a major role in humanoids balance and navigation such as it plays in humans. The Project Humanoid of the University of Aveiro (PHUA) is used as a framework to evolve the thesis of this dissertation and all the mechanical components of the PHUA's neck were rebuilt to guarantee a good infrastructure basis. Image processing and tracking algorithms were developed to find and track a fixed target on an image. Based on the image feedback, a neck's tracking control algorithm was implemented to track the target. The information of the position of the neck may be further used to integrate with other sensor data aiming to improve the PHUA's balance. Throughout the information of the angle of the pan and tilt servomotors and knowing the distance of the target, there were calculated the equations that translate the position of the pan and tilt unit in the world and therefore, the robot's position. The software development is sustained by the Robot Operating System (ROS) framework following the philosophy of a modular and open-ended development. An industrial anthropomorphic robot was used to reproduce the humanoid movements in order to test the whole tracking and ego-motion system. The results showed that the computer vision algorithms present a satisfactory performance for the specific needs and the velocity control algorithm for the tracking system suits the best to accomplish a good and simple tracking system infrastructure in order to obtain the visual feedback for the humanoid.

Contents

List of Figures

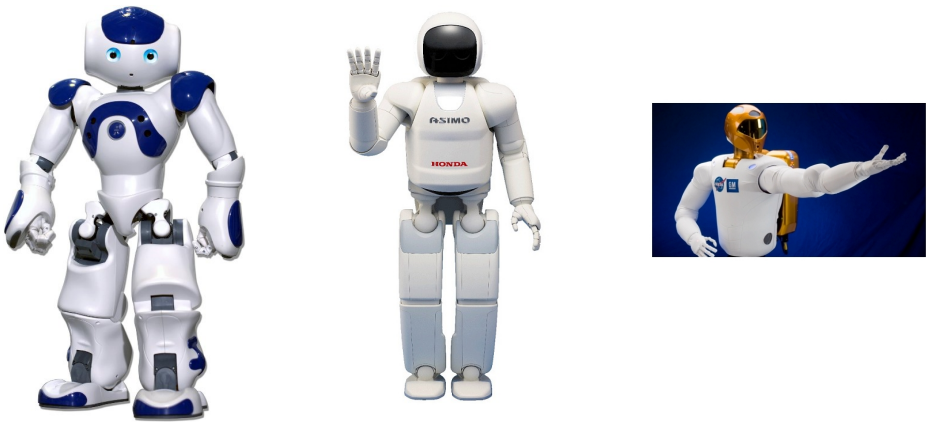
Acronyms

API	Application Programming Interface
ASIMO	Advanced Step in Innovative Mobility
BSD	Berkeley Software Distribution
CAD	Computer-Aided Design
CCD	Charge-Coupled Device
DEM	Department of Mechanical Engineering
DETI	Department of Electronics, Telecommunications and Informatics
DOF	Degree Of Freedom
FOV	Field of View
HSV	Hue Saturation Value
IEETA	Institute of Electronics and Telematics Engineering of Aveiro
INRIA	Institut National de Recherche en Informatique et en Automatique
IMU	Inertial Measurement Unit
LAR	Laboratory of Automation and Robotics
MEAI	Master in Industrial Automation Engineering
MSE	Mean Squared Error
NASA	National Aeronautics and Space Administration
OPENCV	Open Source Computer Vision Library
P	Proportional controller
PD	Proportional Derivative controller
PHUA	Humanoid Project of the University of Aveiro
PID	Proportional-Integral-Derivative
PTU	Pan and Tilt Unit
ROS	Robot Operating System
SP	Set-Point
SVO	Semi-direct Visual Odometry
TCP/IP	Transmission Control Protocol/Internet Protocol
VCR	VestibuloCollic Reflex
ViSP	Visual Servoing Platform
VO	Visual Odometry
ZMP	Zero Moment Point

Chapter 1

Introduction

Humanoid robots might be the revolution of the XXI century [1]. Building a human-like robot has always been the dream and the desire of the human being, leading the most developed countries to invest efforts and money to reach that dream along with the creation of many science-fiction movies [2, 3]. Given the fast development of technology, humanoid robots have become one of the main focus of robotics research and many efforts have been made by several companies and research groups around the world in order to make that dream come true [4]. Since the 70s, many famous humanoid robots have emerged in society, such as ASIMO built by Honda, NAO by Aldebaran Robotics or the NASA's Robonaut (Figure 1.1) [5–10].



a) b) c)

Figure 1.1: Famous humanoid robots a)NAO by Aldebaran Robotics , b)ASIMO by Honda and c) Robonaut by NASA

Due to its biped support, some of the most important and challenging concerns in this area are postural balance and control, either when standing or walking [11] [12]. From a robotics point of view, the maintenance of a stable posture has been

essentially associated with the control of the Zero Moment Point (ZMP)[1], which is a well-known indicator of dynamic stability, but with a reduced use of vestibular signals[2] or vision. A common control strategy is to use modulated feedback of preprogrammed joint trajectories in which the desired motions are calculated in advance using some form of pattern generator formulation (*e.g.*, parameterized curves, optimization of some metric) or capturing human motions by taking recordings (*e.g.*, walking, climbing stairs, etc). These planned trajectories are then played back during walking and modified online through feedback according to a ZMP-based control law.

At the same time, as more information humanoid robots obtain about themselves and the environment around, the better they will adapt to unknown situations and environments [14]. In this context, there has been a significant amount of work in the integration of multiple perception systems based on proprioceptive, inertial and vision sensors, by exploring their complementary characteristics. Multiple sensory sources may allow for more complex sensorimotor strategies, leading to improved robustness across a variety of environments and perturbations.

1.1 Motivation

As for humans and animals, vision is a crucial sense for a robot to interact within its environment. In humanoid robots, vision sensors are often present for the purpose of general exteroceptive perception, supporting behaviors such as gait cycle modulation, navigation and obstacle avoidance. At the same time, it is expected the use of the on-board vision system to assist internal mechanisms of stabilization, as shown by the works of Oda N. *et al.* [15, 16]. In any case, the ability to extract features in dynamically changing environments is one of the key component of robot perceptual systems in order to perform scene classification, object detection and recognition [17]. These features can be used as references to estimate or enhance the knowledge of its ego-motion, *i.e.*, the inference of the rigid motion of the humanoid robot in three dimensions relative to the environment. Egomotion estimation for humanoid robots can help in stabilization tasks, as well as navigation and mapping.

This dissertation addresses the problem of detecting and tracking of static visual cues in the environment and it explores the computational advantages of fixation on a target point by using a camera mounted on an active pan-tilt unit (PTU). The motivation of this work relies on the assumption that, **given the existence of fixed characteristics on the scene, vision may play a major role in estimating the humanoid robot's ego-motion such as it plays in humans.**

[1]Where the total of horizontal inertia and gravity forces equals zero [13].

[2]Sensory system that provides the leading contribution about movement and sense of balance [13].

The work was carried out in the context of an ongoing larger project called Humanoid Project at the University of Aveiro (PHUA) conducted by the Department of Mechanical Engineering (DEM) and the Department of Electronics, Telecommunications and Informatics (DETI) of the University of Aveiro (Laboratory of Automation and Robotics (LAR)). The PHUA aims to provide a multidisciplinary research platform by combining mechanical, electrical and computer engineering for studies in locomotion, navigation and perception in autonomous humanoid robots [18].

1.2 Objectives

The main objectives of this work are twofold. First, to develop techniques that analyze the image for detecting and tracking a static visual cue. Second, to explore fixation on the visual target with a camera mounted on an active PTU by implementing the underlying feedback loop (visual feedback). Fixation is used both to obtain the control signals of the PTU and for investigating the possibility of inferring the camera's ego-motion using images from a single camera. In line with this, the work is developed according the following sub-goals and tasks:

1. Study and familiarization with the problems of visual-feedback and ego-motion estimation, as well as the hardware and software components associated with the work, namely the software development environment under Linux/ROS/C/C++;
2. Extraction and tracking of the selected static visual cue from the image using the OpenCV and ViSP libraries [19][20];
3. Development of the control algorithms to align the head (PTU) with those features (visual feedback);
4. Estimation of the camera's ego-motion based on the information provided by the servomotors encoders;
5. Evaluation of the overall system's performance by attaching the PTU to a Fanuc manipulator in order to obtain a reliable ground truth.

1.3 Dissertation Outline

This dissertation is divided into six chapters: Chapter 2 provides a review of the state of the art about the influence of vision in human's balance and how this concept is being implemented into humanoid robotics. Moreover, it is described what is visual odometry and its relation with robot's ego-motion. This chapter concludes with a presentation of the state of the PHUA and the author's proposed approach in line with the motivation and the objectives of this dissertation.

Chapter 3 introduces a review over PHUA's suitability to implement the proposed objectives and reports the adaptations made in the PHUA's mechanical structure with the implementation of a PTU for the camera. In this chapter, it is described the hardware and software development framework for this dissertation as well.

Chapter 4 describes how the image acquisition was performed and how the image processing and tracking algorithms were implemented as well as the evaluation of its performance. The control algorithms for the PTU are presented. The calculations to get the robot's ego-motion estimation are described and reported in a MATLAB simulator script made for this dissertation to perform some simulations and data analysis.

In Chapter 5 the experimental results are presented and discussed. Finally, Chapter 6 presents the conclusions and open issues to be addressed in future works.

Chapter 2

Framework and Problem Definition

This chapter begins with a brief description of the PHUA project and the main characteristics of the humanoid platform available for research purposes. It follows an overview about the role of the sense of vision for posture and balance in humans and humanoids. At the same time, it is conducted a review of the literature related to the problem of visual odometry and ego-motion. Finally, it is discussed the approach proposed in this dissertation in order to cope with the objectives of the work.

2.1 The PHUA Project

The PHUA project has begun in 2004 with the mechanical design of a humanoid platform. Over the years several master students have been making their dissertations on this project reaching at the moment a humanoid platform, illustrated in Figure 2.1, with 667 mm of height, 6 kg of mass and 27 degrees of freedom (DOF) [21, 22]. The 27 DOF are distributed by 2 passive DOF in the toes and the others active DOF include 12 in the legs, 3 in the trunk, 4 on each arm and 2 in the neck actuated by digital HITEC® servomotors[1] [24].

The humanoid platform includes a rich variety of sensors such as joint and inertial sensors along the overall structure, force sensors in the feet and the vision system in the head [25]. The feet of the robot are equipped with four miniature load cells placed on the feet's corners. The hardware architecture is prepared to accommodate a network of inertial devices (up to 9-axes) distributed along the robot structure. Currently, vision is assured by two cameras with distinct focal lenses for active foveated vision. To allow for both a wide field of view and high resolution vision, there are two FireWire cameras (Pointgrey Firefly MV) attached to the system. One camera captures a wide-angle view

[1] A servomotor is a rotary actuator that allows for precise control of angular position, velocity and acceleration [23].

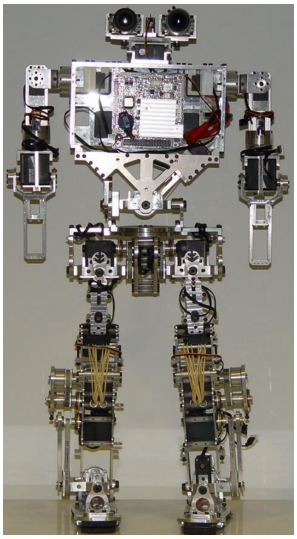


Figure 2.1: Humanoid Project of the University of Aveiro.

of the periphery and the other captures a narrow-angle view of the central (foveal) area, supported by lens with focal lengths of $8\,mm$ and $3.5\,mm$, respectively.

In respect of the PHUA's balance, to approach a concept for robot learning by demonstration, Cruz P. [24] has started to implement an haptic interface using force feedback for teleoperation which is still on course. With the purpose of giving more information to the robot about its internal references and intra-dynamics, an Inertial Measurement Unit (IMU) platform consisting of accelerometers, gyroscopes and magnetometers was implemented by Rafeiro T. in 2013 with 9 sensors achieving a total acquisition frequency of $7\,Hz$ [25].

This is an evolving project in which the author aims to give another reliable source of information to the multi-sensory system PHUA; the ability to navigate with the vision sense. With regard to related works in the PHUA, Rodrigues, A. [26] was the only author that has interacted with the vision system of the PHUA so far. He has implemented a computer vision system independent from the whole one that control the humanoid. The system was built with the intention to track a ball with a specific color and assumes that the ball is alway on the floor regarding the computer vision perception. The image was converted to Hue Saturation Value (HSV) image format representation and the ball was identified from the scene based on its color. The employed camera was an FireWire camera UniBrain Fire-i. The image resolution was reduced from the original in order to increase the performance, it was obtained a *framerate* of $\sim 25\,Hz$ [26].

In relation to the kinematics, the calculations to obtain the 3D position of the ball in relation to the global reference of the humanoid robot were made based on the rotation of the PTU and the position of the ball in the image plane. They can be used

in further studies.

The implemented tracking system was grounded on a simple proportional controller based on the position of the ball in the image and the angle of the servomotors of the PTU[2]. No further study about the performance of the system was reported, specially the tracking system. Moreover, the vision system aims to be a powerful source of information for the robot, for such tasks as balance, navigation and obstacles avoidance, modulation of the gait pattern according to the scenario (foot placement) among others.

2.2 The Role of Vision in Posture and Balance

An essential requirement for many motor actions, both in humans and robots, is the ability to maintain balance whether in a static posture or during locomotion. The challenges of balance control lie on the fact that the walking system is not fixed to an inertial reference frame making it difficult to determine its position and orientation relative to the environment. Furthermore, to operate in the real world, humans and robots must be able to cope with uncertain situations, accommodate floor irregularities and react to changes in the environment.

But what is balance? Besides it demands a complex answer given this broad area, it is generally accepted that, as Alexandra S. Pollock *et al.* reported in heath point of view:

> "(...) act of maintaining, achieving or restoring a state of balance during any posture or activity." [27]

2.2.1 Human Balance System

Balance in humans is associated with the input from multiple sensory systems including *vestibular*[3], *somatosensory*[4], and *visual systems* working together. The *cerebellum* in human beings is the organ that coordinates all movements, regulating the muscle tone and the appropriate trajectory. For instance, the impulses for the act of picking up a pen. Those actions are usually made without thinking, working below the level of conscious awareness allowing conscious brain to work without being stressed out [28]. In fact, as Jones G. referred, conscious control can be tragic [29].

The *brain stem* contains centers for visual and auditory reflexes, which enables it to initiate the movement [28]. Balance is also context dependent and there are several

[2]It is based on the angle of the servos because it was assumed that the ball is on the floor.
[3]Sense of spatial orientation [13].
[4]Complex sensory system made up of a number of different receptors [13].

levels of nervous system that must take into account its complexity complexity [29]. Figure 2.2 illustrates the position of the *cerebellum*, *brain stem* and the *cerebral cortex* in a human head (modified from [30]).

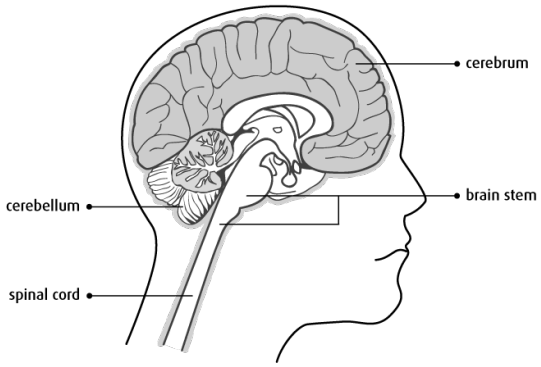


Figure 2.2: Position of cerebellum and cortex in the brain.

The inner ears provide information about gravity and head movements, similar to a gyroscope. Furthermore, humans have a large number of receptors sensing mechanical forces providing information about the positions of body parts in relation to each other [31]. Figure 2.3 represents a diagram of the interactions between the main parts of the cerebrum that are responsible to maintain a postural sway (adapted from [32]).

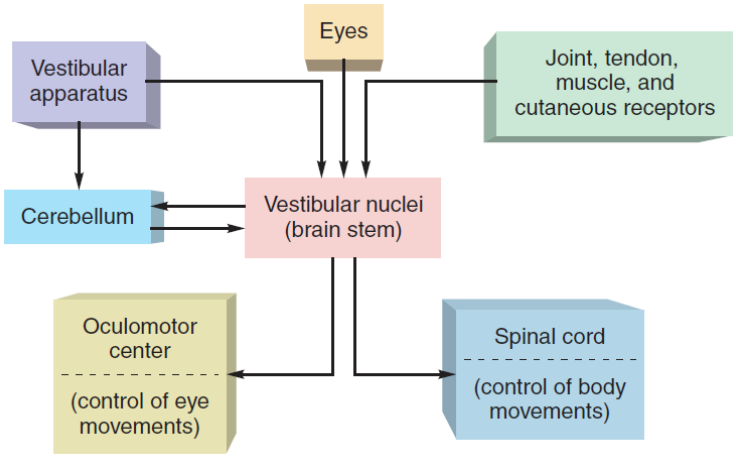
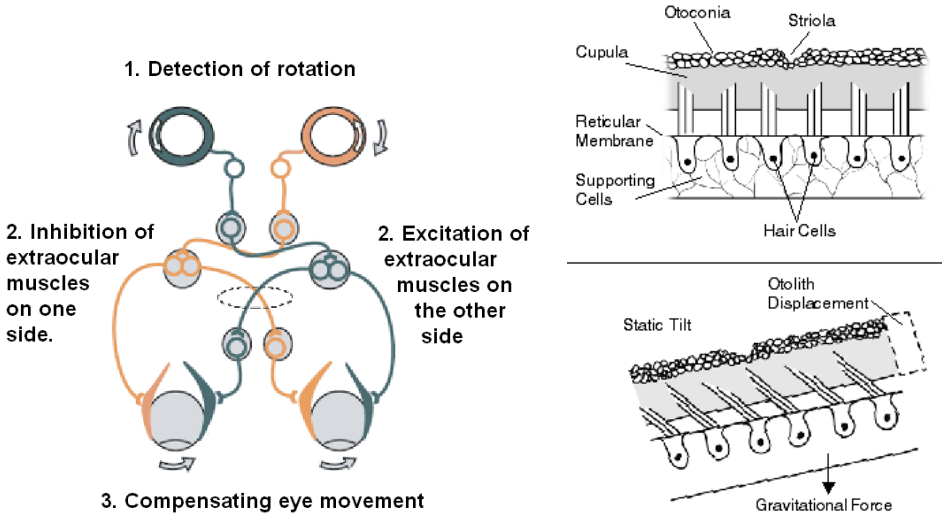


Figure 2.3: Neural pathways involved in the maintenance of equilibrium and balance in humans.

In Figure 2.3 it is verifiable that the vision system, through the eyes, has influence in humans balance. If the scene is slipping across the eyes faster than few degrees per second, the visual system in the brain is too slow to process that information [33]. Therefore, understanding what is on human's field of view (FOV) while moving the head is as important as the movement of the eyes towards the feature to make an

adequate use of the visual ability [34]. The region of the inner ear works with the visual system to keep objects in focus when the head is moving as Figure 2.4 a) suggests.



a) b)

Figure 2.4: Vestibulo-ocular reflex. a) The rotation of the head is compensated by the movement of the eyes. b).

When the head moves, a fluid called *endolymph* moves with the head's inertia causing the movement of the hair cells, transmitting information to the vestibular system, Figure 2.4 b). This is an involuntary process called vestibulo-ocular reflex regulated by the cerebellum (adapted from [13, 34]).

Humans are also able to stabilize the head in relation to the environment. This process is called *vestibulocollic* reflex (VCR), enabling us to better stabilize the *ground truth* for the vestibular system. Consequently, it helps the vision system and the adequate reception of auditory information [35]. Nevertheless, this system is particularly critical in 1–3 Hz range when the head is in resonant instability [36]. For example, a well-known empirically impression of self-movement, is the phenomenon of *vection*, which can be produced by visual stimulation, such as sitting on a stationary train while a neighboring train starts to move. Despite the lack of any change in mechanical forces on the body, the visual motion causes a brief but strong sensation of self-motion [31].

Some animals have better sense of balance than humans. For instance, cats use the tail to play as a counterbalance [37] and most birds eyes, in proportion to their own body size, are much bigger than other animals [38, 39]. This explains why birds do little movements with their eyes. Instead of keeping their eyes focused on the feature such as humans, birds keep the entire head still in relation to the environment. For

instance, the well-known head movement of pigeons while they are walking. Moreover, some birds have the facility to change their *cornea's* shape, enabling them to have a greater zoom range[13, 40].

As we have seen before, it has became evident that vision plays a major role in animal's balance and stabilizing the head or move the eyes help the process of image acquisition. It is also evident that the complexity of this processes goes way beyond of the context of this work, nevertheless, the working principle was demonstrated.

Empirical Experiment

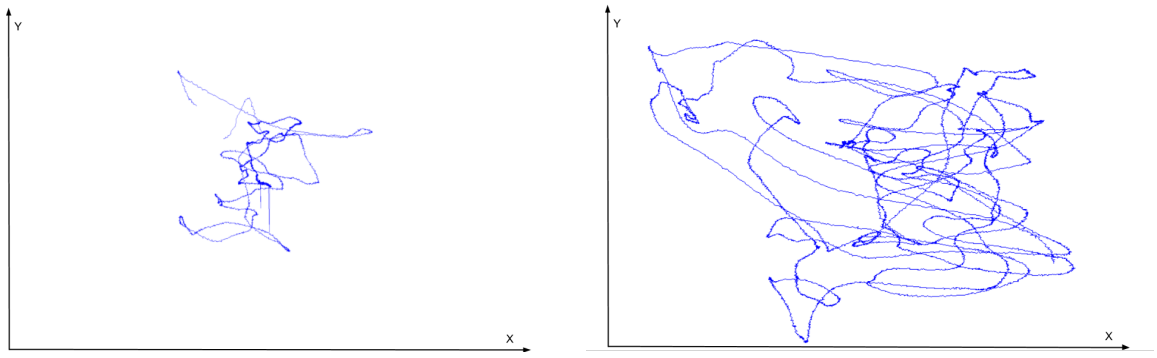
In parallel with this master dissertation, in the context of the Project in Automation Engineering, part of an ongoing collaborative project between the School of Health (ESSUA) and the Institute of Electronics and Telematics Engineering of Aveiro (IEETA), it was made a study to determine the response (or adjustment) in state of equilibrium of a human participant facing difficulties balance positions, with a specific focus on understanding how sensory information from our eyes helps to guide and correct balance (Figure 2.5)[41]. It was also made an experiment using a protocol which includes detailed procedures, as well as the list of required equipment. The experiment comprised a VICON system[5], a force platform to determine the force made by the feet and a head-mounted camera to infer what is the visual information available to subject at every given moment.

Figure 2.5: Empirical experiment at ESSUA department at the VICON's room.

The experimental results are displayed in Figure 2.6 which represents the differences in the position of the Center Of Pressure (COP) with eyes opened a), and eyes closed b)

[5]Motion capture system, based on stereo vision to measure the position of a subject's body.

when the subject is on tiptoes with the legs closed, hindering the subject's equilibrium. Both images are displayed with the same axis scale.



a) b)

Figure 2.6: Comparison of COP on the ground in the experiment with eyes opened a) and closed b) when a human subject is on "spout standing" with the legs closed (same scale).

Preliminary results suggest, as expected, that vision has a major role in postural sway.

Next subsection will focus on robots balance and how the concept of visual-feedback has been introduced to help humanoid's balance.

2.2.2 Vision-Based Balance in Robots

In the introduction of this dissertation was mentioned that balance based on visual-feedback has been introduced in humanoid robots, but the question - how? - remains. Fukuoka, Y. *et al.* introduced this concept in 1999, nevertheless, without success, concluding that visual feedback has a large time delay [42]. Afterwards, in 2006, Ushida, S. *et al.* emphasized the need of gaze following the line of humans, arguing that the visual sensors are placed in the head of the humanoid robots and they swings violently with the robot's motion. Therefore, the quality of the acquired images would improve given to the reduced disturbances that consequently enable them to better tracking [43]. Given the evolution of technology over the last years enabling a better performance of hardware and software systems, the problem that Fukuoka, Y. *et al.* had with the large time delay tended to be suppressed and in 2011 Ushida S. concluded the following results [11, 43]:

> "The complementary or concurrent utilization of the vision with internal sensors provides more reliable control structure of biped walking systems in the future." [15]

The findings need further research. Faragasso A. *et al.* affirm that it is necessary to boost the usage of exteroceptive sensing capabilities given the complexity of these robotic systems [44].

Oda N. *et al.* in 2011, used a monocular Charge-Coupled Device (CCD) camera and a simple visual target to measure the camera's position deviation which is related to the robot's deviation, as Figure 2.7 suggests (adapted from [15]). To get the target position on plane, the algorithm CAMShift, which uses a HSV[6] color feature model, was used to obtain the object position on image [15, 45]. Knowing that $\triangle q_{robt} \propto \triangle v_{image}$, where $\triangle q$ is the variation of the joints and $\triangle v$ is the shift between two successive images, it is possible to extract the sum of the joint deviations. In 2013, the same author has applied optical flow vector[7] for visual feedback control increasing the stability of ZMP [16].

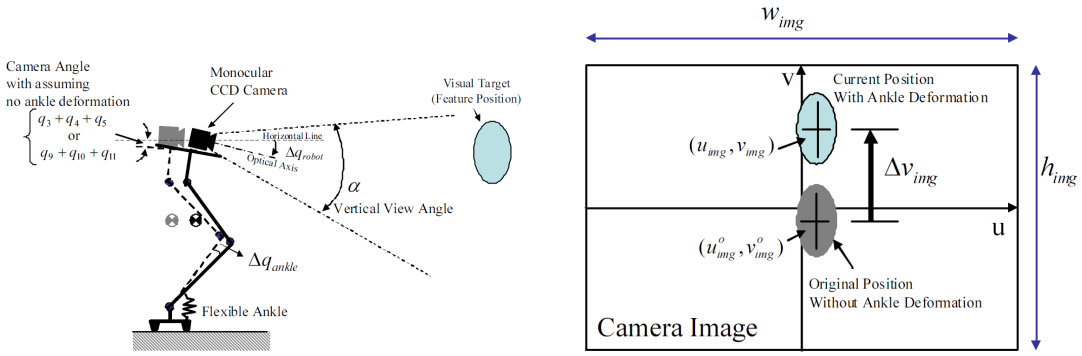


Figure 2.7: a) Visual sensor model (side view) and b) Visual target in camera image.

Regarding the visual perception to extract clues to be used as features to track, a great deal of research has been conducted in the last years given to the fast development of technology in several areas of robotics and automation.

2.3 Visual Odometry

Humans use the vision and the vestibular systems to be able to create a mental map and navigate in practically any environment [46]. In robotics and computer vision, visual odometry (VO), aka ego-motion estimation, regards to the process of incrementally estimating the 3D pose of an agent (*e.g.*, vehicle, robot or camera) relative to the surrounding scene. This estimation process is based on the analysis of the associated

[6]Hue, Saturation and Value.

[7]Pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer [13].

changes in the on-board camera images. In a VO system, the core computational step performed for every image is motion estimation. Two main approaches to compute the relative motion should be mentioned: (1) appearance-based methods that use the intensity information of all the pixels in two successive input images, and (2) feature-based methods that only use salient and repeatable features extracted across the images. The VO research was dominated by NASA/JPL (Mars Exploration Rovers [47]) and the approach has been used in a great deal of projects.

The estimation of 3D motion parameters has been addressed in the past as a reconstruction problem: given a monocular image sequence, the goal was to obtain the relative 3D motion to every scene component as well as a relative depth map of the environment. Most of the VO approaches are based on the construction of an optical flow field, the pattern of relative motion between the camera and the scene. For example, in 2009, Chandraker, M. *et al.* brought in a motion estimation using a stereo pair with straight lines as features and infinite lines for indoor environments taking advantage from the edges [48]. In 2014 Forster C. *et al.* introduced a very precise Semi-Direct Monocular Visual Odometry (SVO) - the fastest in the state of the art - operating directly on pixel intensities and using a probabilistic method.

VO may be complemented with other kind of localization techniques taking advantage of the fact that it does not depend on the locomotion type or the surface conditions [49]. Many studies have been made in order to aggregate other localization approaches such as Inertial Measurement Units (IMUs) and laser odometry [50]. Nowadays, it is common the attempt to fuse different sensor data in order to achieve a good performance at a low cost. For example, the integration of visual and inertial measurements into a coherent ego-motion state seems to be promising for several reasons. First, these sensors are already available in most platforms and this process has accelerated lately with the availability of cheap, small and lightweight sensors. Second, they exhibit complementary characteristics, mainly in what concerns failure modes. On the one hand, the quality and robustness of ego-motion estimates from imaging data depends on the FOV, the spatial distribution of the scene, occlusions, photometric characteristics, illumination among others, but none of them affects inertial measurements. On the other hand, the quality of inertial measurements depends on temperature, gravity, drifts and biases, none of which affects imaging data.

The applicability of these methods to **humanoid robots** requires special attention due to cyclic acceleration spikes occurring during contacts between the feet and the ground, sideway motions during the transfer of support and unexpected perturbations that may occur during walking (*e.g.*, floor irregularities and changing environmental conditions). For the specific case of humanoid robots, a first successful validation of a visual SLAM was presented by Stasse *et al.* in 2006 using the HRP-2 humanoid robot

[51]. The problem of motion blur is addressed in [52] in images acquired by humanoid robots. Authors propose an eqipolar geometry-based visual odometry system whose detection and tracking scheme explicitly models the presence of motion blur.

New approaches have been proposed that do not try to recover a complete motion and structure description. Instead, they try to give individual solutions to tasks where motion is involved such as independent motion detection, time to collision estimation, obstacle detection, convoy following, etc. An exhaustive survey on the best-known vision-based ego-motion estimation techniques can be found elsewhere [49].

2.4 Proposed Approach

This dissertation addresses the problem of detecting and tracking a static visual cue in the environment and it explores the computational advantages of fixation on a target point by using a camera mounted on an active PTU. In this context, fixation means holding the gaze direction towards the same environmental point (target cue) through time. The proposed approach implies the implementation of a visual feedback loop to obtain the control signals for the PTU. In line with this, it was investigated the possibility of inferring the camera's ego-motion using a sequence of images from a monocular system. Some parameters of the camera motion are estimated from the motor encoders relative to the camera's own coordinate system, wherein fixation is used for the sake of simplifying the interpretation of motion.

In this work, the following assumptions and simplifications were considered: First, instead of using the humanoid robot, this work considers only the robot's head constituted by a support base and a camera installed on the PTU. The movement of the robot's head results from fixing the support base to the end-effector of a FANUC industrial manipulator. At this stage, this solution is advantageous because it provides the ground truth information about the robot's motion and, at the same time, allows the exact repetition of the same experiment when changing and evaluation the influence of some parameter on the system's performance.

Second, we assume the active alignment of the humanoid's trunk relative to gravity. This is a fundamental problem faced by legged robots since they are not fixed to an inertial reference frame making it difficult to determine its position and orientation relative to the environment. Accordingly, only translational movements are applied to the robot's head such that the support base remains in a horizontal pose. At the same time, the active PTU involves the control of the degrees-of-freedom (DOF) of the image acquisition, meaning the fixation on a stationary point.

Third, the estimation of some motion parameters requires not only information

from the motor encoders (PTU), but also depth estimation aiming to obtain the distance to the target point. In computer vision and robotics, depth estimation is most commonly done via stereo vision (stereopsis), in which images from two cameras are used to triangulate and estimate distances. Over the past few decades, researchers have developed very good stereo vision systems; see for example Scharstein and Szeliski, 2002 [53] for a review. In this work, these distances are computed by knowing the relative position, subject to additive Gaussian noise, of the FANUC manipulator to the static target point. Anyway, it is worth noting that the performance of the algorithms is tested on real sequences with fixation accomplished during the recording with an active PTU. In future developments, it is expected that stereo vision can be used to estimate the actual distance of the visual cue of interest from the camera.

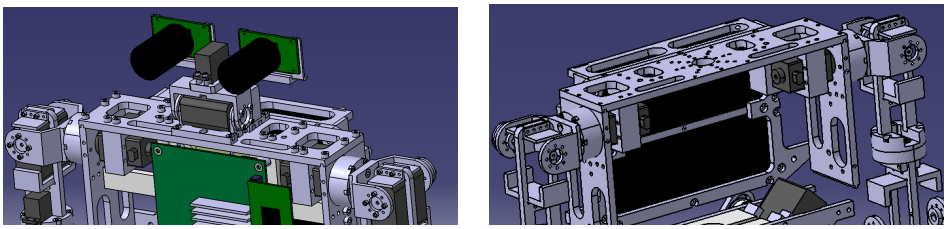
Chapter 3

Hardware and Software Architecture

This chapter conducts a survey on how the needed adaptations to the existent PHUA's structure were performed in order to obtain a reliable mechanical structure to fulfill the objectives. Furthermore, it is described the hardware and software development frameworks as well as the main infrastructure components to accomplish the objectives of this dissertation.

3.1 Neck Structure Design

To ensure the proposed objectives on first chapter of this Dissertation, the first step was to verify the PHUA's mechanical ability to perform a gaze task, which includes the necessary DOF to rotate the neck and a vision system. Given the mechanical structure of the PHUA regarding those parameters, it was verified that the existent mechanical components were damaged and/or under dimensioned for the propose. Figure 3.1 illustrates the state of the art with a Computer-Aided Design (CAD) design illustration at the beginning of this dissertation where the proposed objectives must be performed.



a) b)

Figure 3.1: PHUA's upper body design before the dissertation starts a) with the neck and b) without the neck (hidden).

An assessment of the needs was collected in order to redesign all the PHUA's neck structure, the following points were considered:

- Mechanical restrictions to the existent PHUA structure, camera and servomotors size;
- Put some parts of the neck inside of the truck, reducing the moment induced in the humanoid robot and augmenting its compactness;
- Design the mechanical parts considering the manufacturing process, in order to be feasible to build;
- The FOV of the robot and therefore, maximum rotation of the motors;
- The alignment of the rotational axis of servomotors;
- The possibility of using two cameras (stereo vision);
- The function of each mechanical part to select the components materials;
- To be compact and robust.

After considering several configurations, it was achieved the one represented on Figure 3.2, in this case, only one camera is represented in the middle center.

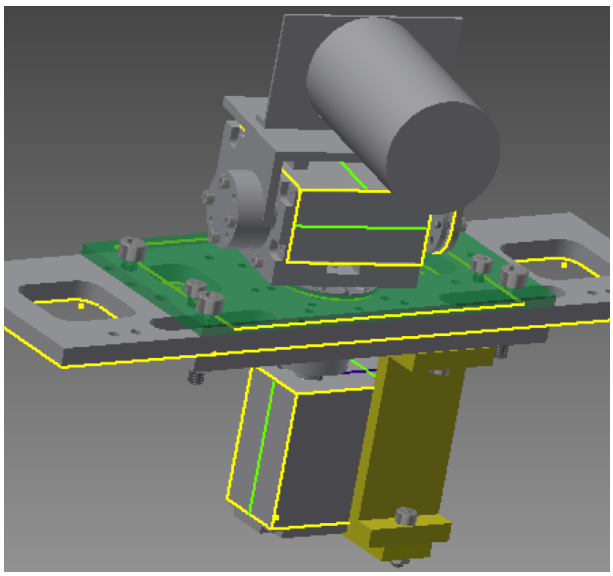


Figure 3.2: PHUA's neck Computer-aided design with software Inventor®.

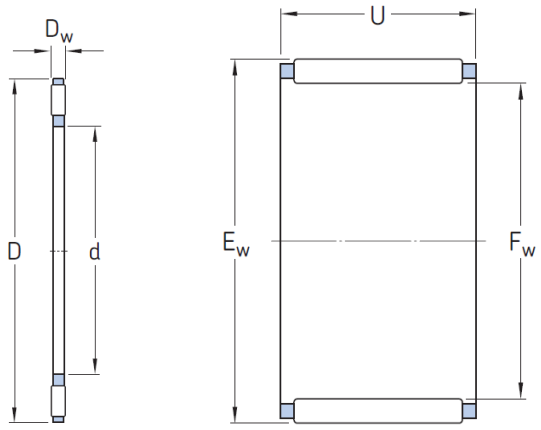
The *pan and tilt* configuration was maintained comparing with the previous structure, because it only combines two DOF's with the possibility to move in 3 dimensions (X,Y,Z). Regarding the FOV, due to the fact that this is a humam-like-robot, it was considered the FOV of a human. Therefore, it was assumed 180^o degrees of range for the *pan* movement and 90^o degrees down for the *tilt* movement so that the robot is able to see its toes, and nearly 30^o degrees up. It is not planned that the robot will look above its head in the future of the project, thus it is not necessary

to increase its dimension (height). Nevertheless, the structure can be easily modified (changing one component) to increase the openness of the *tilt* movement.

The *pan* servo is built-in on PHUA's trunk to increase its compactness, the axis of the two servos are aligned to decrease the moment on the *pan* servo. It was also considered the available mechanical connections with the servomotors to adjust some parameters in the drawings.

Regarding the selection of the mechanical components, the bearings play a major role to guarantee a correct alignment of the moving parts. Some aspects such as the compactness were considered, therefore, it was only applied *needle rollers* requiring the cage to be included in the designed mechanical parts. The selected bearing is a two axial needle bearing guaranteeing the correct alignment of the *pan* spindle to the *tilt* module and the *tilt* movement with the camera, k 12 15 10 TN and AXK 1528 respectively, the other one is a radial needle bearing to support the *tilt* module, K 4*7*7 TN.

Given the size of the cage in the came for the *pan* movement component, it can not be used a combined bearing, because is specially designed for that purpose. Moreover, all the bearings are needle bearings given its compactness. All of those bearings are provided by the company SKF®. Figure 3.3 illustrates how the bearings were selected based on the function and the conditions where were applied (adapted from [54]).



a) b)

Figure 3.3: Types of bearings applied in PHUA's neck, a) axial load b) radial load .

To calculate the two required servomotors torque, it was considered the moment created by the camera, which is the weight that it has to support (the characteristics of the camera, including weight ($120\,g$) are described on the next Subsection 3.2.2) and the distance to the center of rotation. The following equations were applied as presented bellow:

$$g \times m_{camara} \times d = T \tag{3.1}$$

$$10\,(m * s^{-2}) \times 0,12\,(Kg) \times 0,045\,(m) = 0,054\,Nm \tag{3.2}$$

$$0,054 \times 4_{safty\,coef.} = 21,6\,Ncm \tag{3.3}$$

It was considered that both servomotors have the same torque to support, given the weight that they have to support and the distance to the center of rotation, therefore, the requirements are the same. The following Subsection 3.2.1 reports the characteristics of the employed servomotors for the PTU.

In Figure 3.4 is a photograph taken after all components were machined and assembled at DEM workshop.

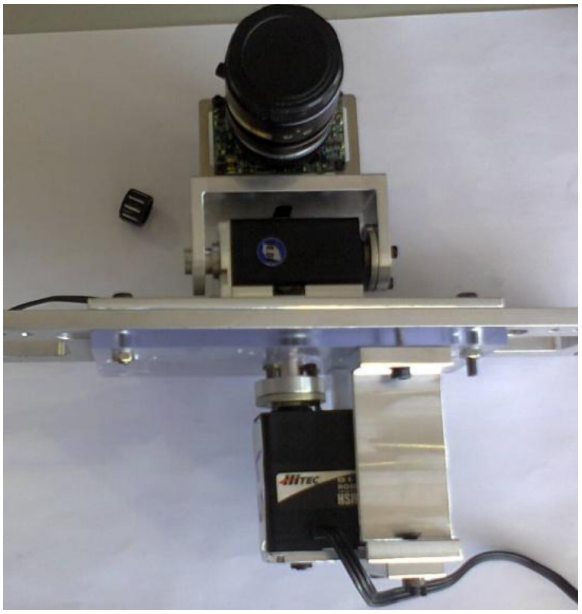


Figure 3.4: Final neck structure after assembly.

In Appendix A are presented all the technical drawings of the designed neck's mechanical parts.

3.2 Hardware Framework

The main blocks that constitute the infrastructure with regard to the hardware are the camera, the servomotors, the central processing unit (computer), the power supply for the DC servomotors and the FireWire and $RS-232$ adapters. Figure 3.5 represents the main constituent blocks and its interactions between each others.

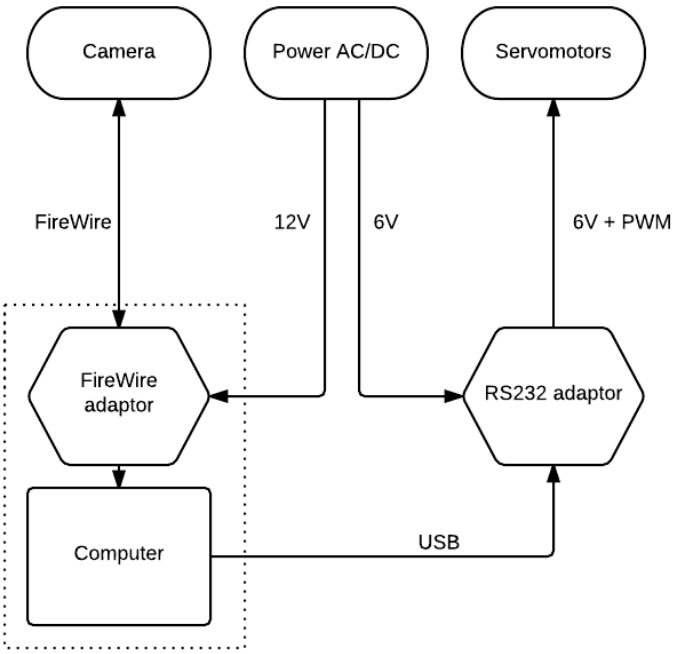


Figure 3.5: Schematic of the hardware connections of the developed system.

The ensuing subsections report the main hardware components of the project, including the camera, servomotors, computer processor and the robot Fanuc.

3.2.1 PTU's Servomotors

In line with the whole PHUA that has digital HITEC® servomotors in all active joints of the humanoid, the ones that were selected should have the same communication protocol to be easily integrated in the PHUA's infrastructure, since all back-end functions are implemented and tested; moreover, it should be noted that the HITEC® brand is a well-known servomotors supplier, producing specific oriented servomotors for robots.

Considering the necessary torque calculated and all the project parameters, including the reduced size/weight, the servomotors selected to control the neck are both $HSR-5498SG$ [55].

The servomotors have an operating voltage range from $6\,V$ to $7\,V$, $0.22\,s/60^o$, $11\,Kg.cm$, steel gears and 180^o of amplitude. The servomotors were made to receive position set-points (SP), nevertheless they allow to change the SP during the trajectory. The input position must be an integer ranging between 600 to 2400, i.e. it is possible to configure the position of the servomotor within a resolution of $0,1^o$. With regard to the velocity value, it is defined in a integer range from 1 to 255 (average velocity) [24].

The servomotor has an internal micro-controller that regulates the PD controller, the micro-controller can be configured with the software supplied by HITEC® [24, 56].

The communication between the servomotors and the central control unit is made by a Bidirectional Serial Interface $RS-232$ within a proprietary protocol called *Hitec Multi-Protocol Interface (HMI) [24].* Figure 3.6 illustrates the interface board between the computer and the servomotors, as well as the power input (on the bottom of the Figure) (adapted from [24]).

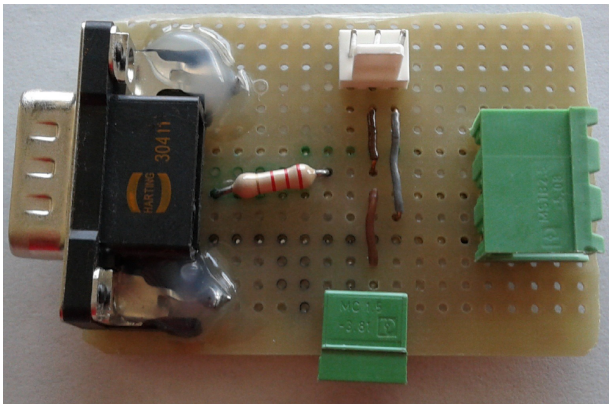


Figure 3.6: Servomotor communication converter board.

For all the servomotors it was attributed an unique ID that may be configured (2 bytes). In this way, it is possible so send specific commands to each servomotor.

3.2.2 Image Acquisition

The image acquisition was made with a camera predestined in the beginning of this project. It is a color CCD camera, FFMV-03MTC made by Pointgrey® with $0,3\,Mp$ of resolution, $120\,g$ of weight, 60 *fps,* and embodies a FireWire $1394a$ communication protocol [57]. The video format of image acquisition is a 640×480 mono8 of resolution and the *shutter* is global (likely to reduce the *motion blur* effect). With regard to the connection of the camera to the Central Processing Unit, it was used the Express card FireWire adapter represented in Figure 3.7(adapted from([1])). The adapter needs an external power source of $5\,V_{DC}$.

The camera's lenses and the camera on its own apply a distortion in the acquired image by the camera such as the distortions by the irregular parallelism of the lenses, the pixel form factor of the camera, the radial distortion of the lenses, among other distortions commonly found in all cameras. Furthermore, in order to get the conditions for the implementation of the computer vision algorithms, it is necessary to make a

[1]http://www.ioi.com.tw/products/proddetail.aspx?ProdID=1060089&HostID=2009

Figure 3.7: Express card FireWire - Unibrain®

software correction of the distortion. One of the big advantages to work on a common used open-source framework with a large online community, the Robot Operating System (ROS[2]), is that many code software are available online, with many general purpose packages including the camera calibration package.

Following the instructions of the reference [58] provided by ROS[3], it was used a chessboard (see Figure 3.8) to make the image's calibration and obtain the calibration file. The captured image frames published[4] by the camera capture node are calibrated in this node. Consequently, those frames are subscribed by the computer vision nodes, performing the computation. This process of calibration is encapsulated in the ROS node *image grabber*, publishing properly calibrated images to other subscribers. This node gets the calibration parameters from the file generated during the calibration process.

With regard to the processor unit that processes the developed algorithms and records the data results over the experiments, it was used a TOSHIBA Satellite A300 computer with the processor Pentium(R) Dual-Core CPU T4200 @ 2.00GHz.

3.2.3 Fanuc Robot

The simulation of the PTU's base movements was carried out with the assistance of an industrial robot available on the LAR where the experiments were performed. The industrial robot used to develop the experiments was an anthropomorphic six DOF robot - Fanuc M-6iB6s - represented on Figure 3.9, capable of manipulating up to 6kg of payload [59]. Since the robot Fanuc is calibrated and has a repeatability of 0,08mm [60], the robot was considered as the ground truth for all distance measurements and movements in all tests, concerning the visual target and the PTU. Moreover, it allows the reproduction of recorded trajectories and different types of movements such as

[2]A detailed description about this software is reported in the next Subsection 3.3.1.
[3]http://wiki.ros.org/camera_calibration.
[4]In the context of ROS, it means that other ROS nodes can subscribe that information.

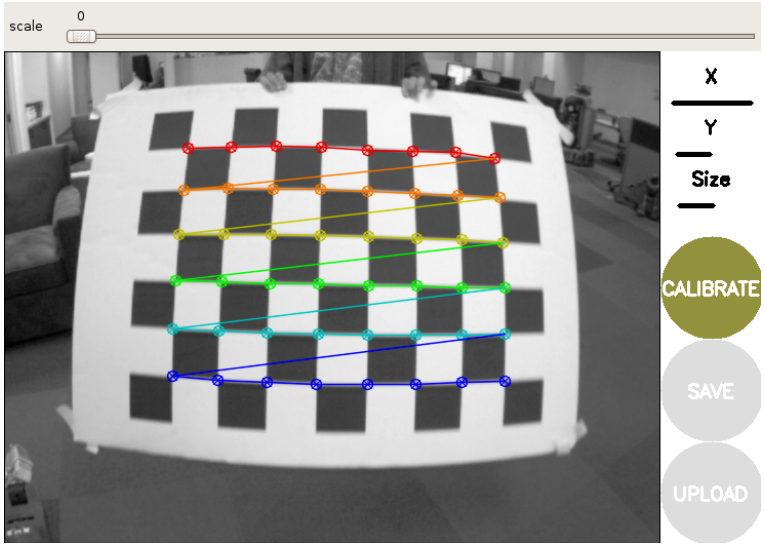


Figure 3.8: *Print screen* of the camera calibration process using ROS *camera_calibration* tool.

linear movements.

Figure 3.9: Fanuc M-6iB6s used for the experiments.

A C++ client application was developed to acquire the data from the robot and control its movements through a TCP/IP[5] connection. With an easy interface that can be further used in other projects and contexts, the application also has the capability to save the data of the robot's end-effector position during the movement on a .csv[6]

[5]Transmission Control Protocol/Internet Protocol.
[6]Comma-separated values.

file; choose the program to load (saved on the robot), data acquisition velocity, or set to receive an external trigger throughout a ROS message to start running. This last mentioned ROS message function is particularly interesting when it is expected to initialize several programs at the same time.

The robot has some preprogrammed commands that can be used to interact with, such as `GETCRCPOS, RUNTPP or SETREG` that acquires information related to the end-effector position, runs a specific previous saved program on the robot memory and set the end-effector position respectively. Finally, this application also concatenates the time-stamps with the recorded robot's position data that can be later used to calculate the speed.

Figure 3.10 illustrates the ROS nodes and the message interactions during the tests. In order to trigger the experiment synchronously, it was used the PC's keyboard which consequently triggers an ROS message. Therefore, this message initializes the FANUC movement and the data recording of its position, moreover, the computer vision messages data and the instructions given to the servomotors gathers information to further study and analyze the experiment. The time of computer controller was used as a reference to synchronize all the recorded data. All the recorded data was saved and converted into .csv files for further analysis with a MATLAB script.

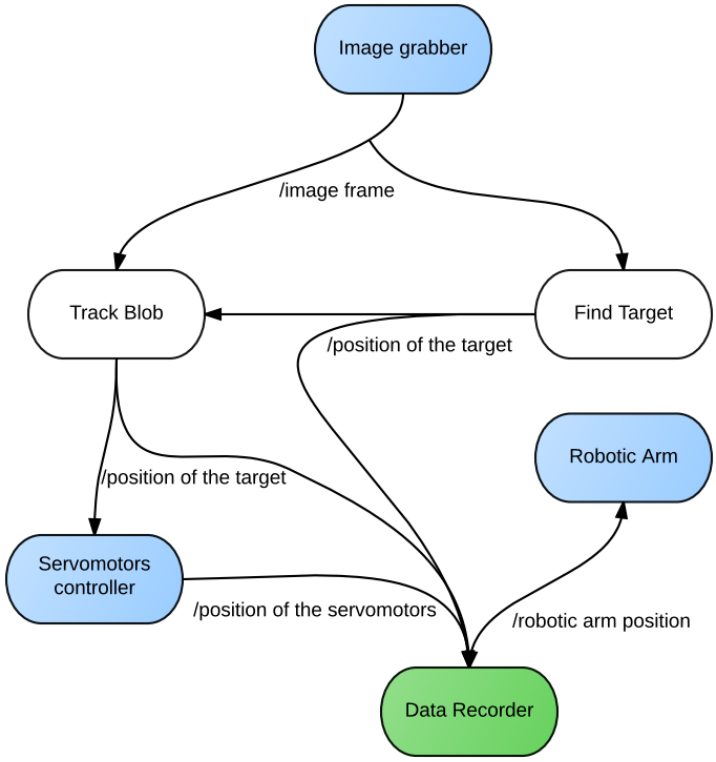


Figure 3.10: Schematic of the software architecture for the experimental setup to test the developed system.

The coming section reports the main software bases that assisted the infrastructure.

3.3 Software Development Environment

With regard to the implemented software, this section presents the main parts needed to build the infrastructure. All the developed software was implemented above the Linux operating system - Ubuntu - version 12.04 - LTS which is the operating system used in the most recent developments of the PHUA and in the whole LAR.

LAR has a repository in which all developed software throughout several projects is available. The software developed in this work will be distributed into two folders of the repository of the LAR. The vision system will be located in the folder *arrow_detection* within the folder *perception* and the PTU control system inside the folder *humanoid_control*, in turn, inside the folder *bases*.

The following subsections report a description about the three main softwares used to build the setup, which are the ROS, OpenCV and ViSP.

3.3.1 ROS

Alike it was mentioned before, this dissertation belongs to an evolving project, in which every contributor that takes part of the project must consider the current state of the work. Therefore, in order to plan their interventions, they must give modularity to the project, enabling the reutilization of the code from other sources and the integration of new functionalities; in this way, the Robot Operation System software has been used in the PHUA since the beginning of the project.

ROS is a free and open source software framework for robot software development. It is widely used in robotics research in which has a large on-line support community, as well as used in many of the most famous platforms such as NAO and Robonaut 2 [5, 6]. Introducing the ROS to the research community, the authors stated the following statement [61]:

> "ROS is not an operating system in the traditional sense of process management and scheduling; rather, it provides a structured communications layer above the host operating systems of a heterogenous compute cluster." [61]

The ROS was originally developed in 2007 under the name *Switchyard* by the Stanford Artificial Intelligence Laboratory. The ROS is lightweight and robust that eases modular software development; provides standard operating system services such as hardware abstraction, low-level device control, implementation of commonly

used functionalities, message-passing between processes, package management, among others [62].

Among many functionalities, ROS is designed to run under Unix/Linux and supports the programing languages *C++* and *Python*, furthermore, it permits peer-to-peer communication between different modules and supply tools for code implementation and testing [56]. Each module (*package*) is related with the group of programs and/or libraries that perform of a specified function (thereof the modularity of the ROS). A *package* may have many nodes that perform computation and they may communicate among themselves through messages [61]. The messages, according to the information they contain, may vary from different types of data, such as *integer*, *character* or complex structures, for example: *cloud points*. The messages are identified by the runtime structure of ROS as *topics*. A node that publishes data on a *topic* is called *publisher*, likewise, the *receiver* is entitled as the node *subscriber* [62].

One of the tools that ROS provides is the *rosbag* which enables to store information of *topics* in *bags* and play them back anytime, allowing them to process and analyze off-line data. A *rosbag* is an API[7] to manipulate and use *bags* for reading, writing, among other options. A *bag* is a file format in ROS for storing ROS message data [63]. The *roslaunch* is another useful tool that permit to launch one or several nodes or even navigation system commands. The following piece of code is an example of the command lines to throw the *ROS master* (1^{st} *line*), launch the camera application and its configurations saved on the configuration file in the package *arrow_detection* $\left(2^{nd}\, line\right)$ and run the image viewer of the camera subscribing the topic */camera/image_raw* $\left(3^{rd}\, line\right)$.

```
1 roscore
2 roslaunch arrow_detection camera.launch --screen
3 rosrun image_view image_view image:=/camera/image_raw
```

Each command must be provided in different command windows; in this dissertation it was used the program *terminator*[8], which allows to display multiple command windows in the same interface.

The ROS version employed in this work was the *ROS Hydro* which is the seventh release version[9]. Besides what was refereed in this section, the ROS provides other functionalities and features that does not apply in this dissertation and will be not reported here. In order to acquire the images from the camera as well as the control of the servomotors, the firmware is available from previous projects of the LAR and the

[7] Application Programming Interface.
[8] https://apps.ubuntu.com/cat/applications/precise/terminator/
[9] http://wiki.ros.org/hydro

online community, underpinning the software structure to develop of the solution.

3.3.2 OpenCV

The OpenCV[10] is an open-source BSD[11] licensed library that includes several hundreds of computer vision algorithms widely used in computer vision applications. It was originally developed by Intel in 1998 before being publicly released in 2000, reaching now a total of more than nine million downloads [64].

The OpenCV stands for *Open Source Computer Vision Library*, it is written in *C, C++, python and Java* and runs under Linux, Microsoft Windows, among other operating systems. Designed to have a good computational efficiency with focus on real time applications and created to rapidly build reasonably sophisticated computer vision systems; the OpenCV combines low-level image-processing functions and high-level algorithms that are extensively used by the research communities around the world. The contents of the OpenCV covers a huge range of applications within hundreds of functions available; moreover, it is well documented with a large online user community. The OpenCV disposes of a vast range of image processing algorithms, some are used in the developed computer vision algorithms to built the infrastructure described in this dissertation (further described in Chapter 5) [64, 65].

In order to integrate the OpenCV with the ROS, allowing the image processing algorithms access to the data published by the camera in the ROS *topic*, the ROS provides the API *cv_bridge,* which is an interface between the ROS *sensor_msgs/Image* message format and the OpenCV, *cv::Mat* data structure. Figure 3.11 illustrates this process[12].

[10] http://opencv.org/
[11] Berkeley Software Distribution.
[12] Adapted from http://wiki.ros.org/cv_bridge/Tutorials/

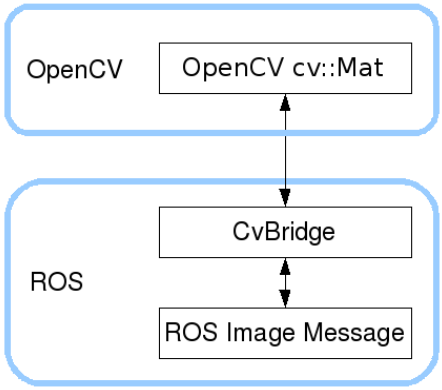


Figure 3.11: Schematic of the interface between the image format of the ROS and the OpenCV, cv_bridge.

The Algorithm 3.1 is the main necessary code to implement the cv_bridge. In this example is subscribed the topic */camera/image_rect_color* taking advantage of the function *image_tranport* and then, when the callback is 'called', the image frame received is converted with the aid of the function *cv_bridge::toCvCopy*. The image data may be acceded further by the variable *cv_ptr*.

Algorithm 3.1 Code example of the implementation of the *cv_bridge*.

```
#include <image_transport/image_transport.h>
#include <cv_bridge/cv_bridge.h>

class ImageConverter {
     ros::NodeHandle nh_;
     image_transport::ImageTransport it_;
     image_transport::Subscriber image_sub_;
public:
     ImageConverter():  it_(nh_) { //CC
          image_sub_ = it_.subscribe("/camera/image_rect_color",1,
                    &ImageConverter::imageCallback, this);
     }
     ~ImageConverter() {}
     void imageCallback(const sensor_msgs::ImageConstPtr& msg) {
          cv_bridge::CvImagePtr cv_ptr;
          cv_ptr = cv_bridge::toCvCopy(msg,
                    sensor_msgs::image_encodings::BGR8);
     }
};
```

3.3.3 ViSP

With regard to the implementation of a robust open-source visual tracking software for the visual tracking, the ViSP[13] was implemented in this dissertation to add value for the computer vision algorithms.

The ViSP states for *Visual Servoing Platform* and is a software platform that permits to build high-level applications for visual servoing[14]. It was created to bridge the gap that existed in software environments that allow fast prototyping of visual servoing tasks, although not dedicated to a certain type of robotic systems, i.e. independent from the hardware [20]. The ViSP is a modular cross platform library implemented in *C++* under Linux, allows the usage of one or several cameras in order to control the motion of a robotic system (visual servoing [66]) aiming to be extensible, simple and cross-platform [20].

The library of the ViSP platform is divided into three main modules which are the tracking, the image processing and further computer vision algorithms, a module for servoing control laws and another for visualization. Figure 3.12 is a schematic diagram of the architecture of the software library [20].

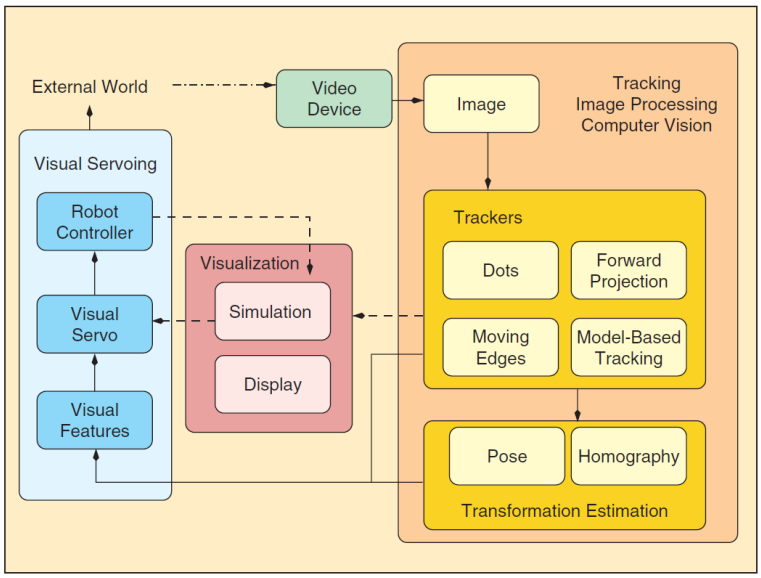


Figure 3.12: ViSP software architecture.

In the context of this dissertation, the purpose of implementing ViSP is to provide the capability to track visual targets (fiducial markers) to the humanoid. The applied ViSP blob tracking algorithm is one of the algorithms that ViSP has available[15]. This

[13] http://visp.inria.fr/
[14] Specification of a task as a control of visual features [20, 66].
[15] Chapter 4 explains this choice.

algorithm uses in the first place the previous *center of gravity* position of the target from the previous iteration. Consequently it detects the *boundary* of the *blob* going right, then follows the *boundary* in order to compute the *center of gravity*, *size* and *moments* from the Freeman chain elements. If the *blob* is not found, checks if it has the same characteristics of the previous iteration. If not, search in the whole image [67].

The *Freeman Chain* is one of the shape representations[16] that is used to represent a *boundary* through a connected sequence of straight line segments of a specified length and direction. The direction of each segment is coded by a numbering scheme in a unified way to analyze the shape of the *boundary*. *Chain Code* follows the *contour* in counter clockwise and keeps track of the directions from one *contour* pixel to the following one [68, 69]. The $C++$ code example 3.2 demonstrates the basic commands to implement a *tracking blob* algorithm using ViSP[17].

Algorithm 3.2 Code example of the ViSP *tracking blob*.

```
#include <visp/vpDisplayX.h>
#include <visp/vpDot2.h>

int main(int argc,char**argv) {
     vpImage<unsigned char> I; //create image
     //config image grabber 'g' | e.g.:  vp1394TwoGrabber
     g.open(I);
     g.acquire(I);//refresh image
     vpDot2 blob; //create blob
     //set blog characteristics
     blob.initTracking(I,vpImagePoint(target_pos.y, target_pos.x));
     while(true) {
          g.acquire(I);//refresh image
          blob.track(I);//Tracking
          vpDisplay::flush(I);
     }
     return 0;
}
```

Regarding the interaction of ViSP with the ROS, ViSP provides a library called *visp_ros* that enables the use of ROS in a transparent way benefitting from ROS

[16] Typically represented based on 4 or 8 connectivity segments.

[17] Adapted from: `http://visp-doc.inria.fr/doxygen/visp-2.8.0/tutorial-tracking-blob.html#tracking_blob_tracking`

features such as Figure 3.13 suggests[18] (similar to the ROS/OpenCV *cv_bridge*) (adapted from [20]). In this way, it is possible to completely separate the code that is specific to the material (frame grabber, robot), from the one that perform the visual servoing.

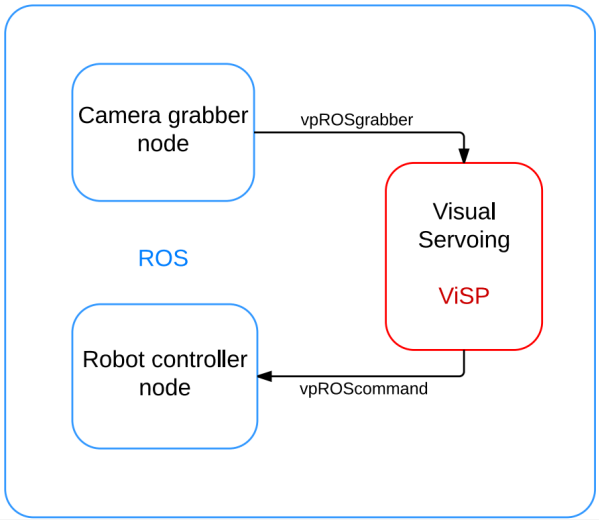


Figure 3.13: *Visp_ros* schematic.

With the *visp_ros,* it is possible to access the information published in the ROS *topics* such as the image frames published by the camera node. The algorithm 3.3 exemplifies the code structure for ViSP to grab the current frame from the ROS *topic image_raw* (adapted from[20]).

Algorithm 3.3 Code example of a *visp_ros* node.

```
#include <visp/vpImage.h>
#include <visp_ros/vpROSGrabber.h>

int main(int argc, char **argv) {
     vpImage<unsigned char> I;
     vpROSGrabber g;
     g.setImageTopic("/camera/image_raw");
     g.open(argc, argv);
     while(1) { //Loop
          g.acquire(I);
          // Do some stuff (e.g.  publish a ROS message an a topic)
     }
}
```

[18] http://wiki.ros.org/visp_ros

ViSP is distributed under an Open Source license and it is developed within the INRIA[19] Lagadic project in France[20]. In this dissertation, it was used the version 2.8.0 of the ViSP library[21]. The installation of the ViSP was made with the following command:

```
1  sudo apt-get install ros-hydro-visp
```

3.4 Final Remarks

As it was refereed before, the PHUA is an evolving project that implies that every new modification must consider the past and the vision of the future, in order to guarantee its evolution towards an autonomous humanoid robot.

Regarding the mechanical structure, it was concluded in one of the previous sections that a reliable neck structure is required to make an effective gaze control. Given this fact, the author proposes to rebuild the mechanical structure of the PTU, in order to guarantee a good ground truth and to avoid problems in the future.

In this chapter was described how the setup was refurbished to accommodate the needs. This functional and stable setup makes possible to study and make the proof of concept of the computer vision, control and ego-motion algorithms. Figure 3.14 presents a schematic of the modules that constitute the proposed approach. The developed algorithms and the discussion are described on the following chapters.

[19]Institut National de Recherche en Informatique et en Automatique.
[20]http://www.irisa.fr/lagadic/visp
[21]http://visp-doc.inria.fr/doxygen/visp-2.8.0/

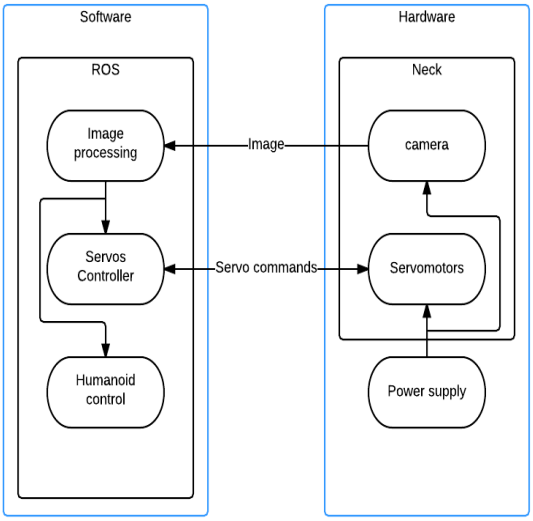


Figure 3.14: Schematic of the proposed approach

Chapter 4

Vision-Based Robotic System

This dissertation proposes that an humanoid robot may benefit from fixed existent characteristics in the environment around. As a result, the robot may have the capability to extract those features throughout its visual system, interpret them and make better decisions with that information.

Considering that the previous chapter describes how the setup was assembled to test the hypothesis of this dissertation, this chapter presents how the image processing was carried out in order to extract those features. Those features will give information to instruct the PTU's servomotors and add valuable data to the PHUA, namely the navigation system.

The first section reports the target characteristics that will represent the scenario and the developed computer vision algorithms (image processing and tracking). After the performance of the implemented computer vision software is characterized. The second Section presents the control algorithms to command the PTU's servomotors. The third section describes the calculation in order to obtain the ego-motion equations based on the PTU's servomotors angles and the distance to the target. Finally, the developed MATLAB PTU's simulator and some conclusions are described.

4.1 Vision System

Instead of making a wide research on how to extract fixed characteristics in any scenario, the purpose of this dissertation is to select a representative one that will simplify the vision feature extraction process. Simultaneously, it gives the author the possibility to learn more about computer vision, which is one of the main objectives of the dissertation, as a subject from the Master of Industrial Automation Engineering (MEAI).

Figure 4.1 is a schematic summarization of the ROS nodes and its interactions with the computer vision implemented software, described in this chapter.

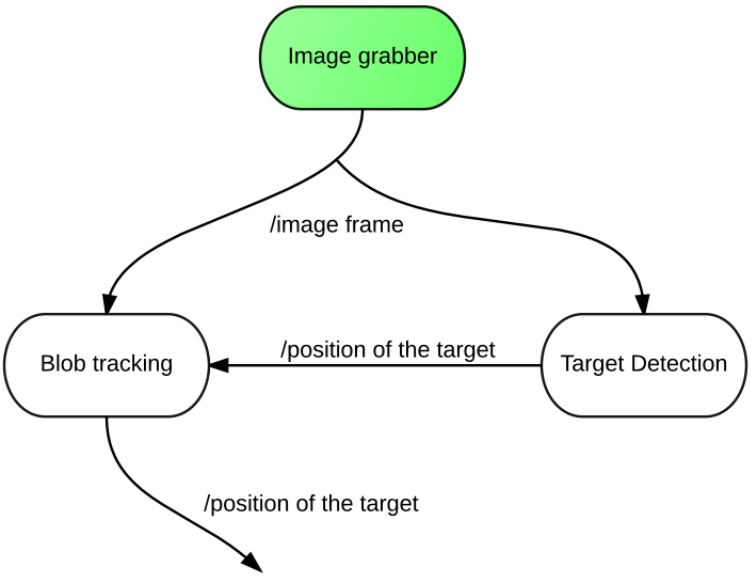


Figure 4.1: Graphical representation of the ROS regarding the image processing.

The *image grabber* is the node responsible for grabbing the images from the camera and publish each frame in a image frame topic called */image_raw.* The node called *find target* subscribes the topic */image_raw*, finds the position of the target's center in the image and publishes a topic called */find_arrow_position* where a message with the center point of the target in the frame is given. The *track blob* node subscribes both the */image_raw* topic and the */find_arrow_position,* and using the ViSP library, applies an algorithm to track the blob located in the image at the position given by the */find_arrow_position* message. After computing this information, this node publishes the SP with the information of the target distance to the center in a topic called */find_arrow_state* (to the humanoid controller).

The ROS messages that contain coordinates of an image in the implemented software code are of the type *geometry_msgs/Point*, exploiting the advantage of the ROS built-in common messages.

The selection of the target as a reference to extract the humanoid's posture[1] was based on a major reference for the work reported in this chapter, which is the research made by Oda N. *et. al.* [15]. This authors have selected a simple target's shape to be used as a reference for the tracking system (fiducial marker), thus, the selected one is an arrow shape target as Figure 4.2 illustrates.

[1]Fixed characteristic on the scene.

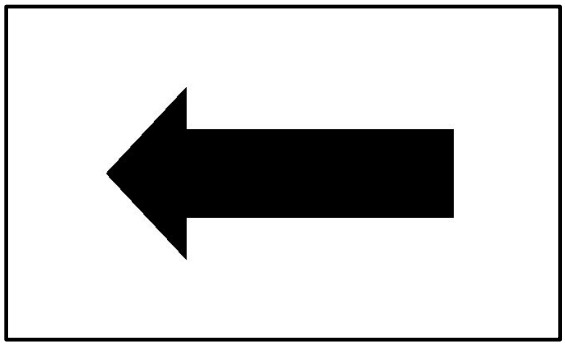


Figure 4.2: Target used as reference for the visual system.

An arrow shape is invariant to rotation and still, as well as fairly easy to extract its dimension. Moreover, it has a relatively huge blob area to track and it is not so common on a LAR's controlled environment, by the fact that it may be confused with other background noises. Next section explains the algorithm in order to find the target on the PTU's camera image.

4.1.1 Target Detection

To decode the target position on image, considering that the target is a black blob on a white board, it was applied the following pre-processing algorithms: (1) conversion of the image data to a gray-scale image format, (2) application of a smoothing algorithm to eliminate some background noises, (3) usage of an adaptive threshold and a closing morphology to transform the gray-scale image on a black and white image and delete small regions.

The consecutive stage was to identify the blobs on the image and filter which one of those was the target (if it exists on the image!). Therefore, it was applied the OpenCV function*, findContours,* with the goal to extract all the contours of the image blobs and then, filter them with the following ordered criteria with regard to the specific characteristics of a target's shape:

1. The number of line segments (the target has seven segments);
2. The area of the blob;
3. The perimeter of the contour;
4. The relation between the width and height of the smallest bonding box (it is considered that the target must always be relatively parallel to the image plane);
5. The number of segments of the contour with the *convex hull* function(must be 5 segments).

The order of the previous criteria is very important for the algorithm's performance; for instance, the *number of line segments* must be the first, by the fact that is very fast to compute and eliminate many unwished blobs. Figure 4.3 illustrates the process to find the target on the image.

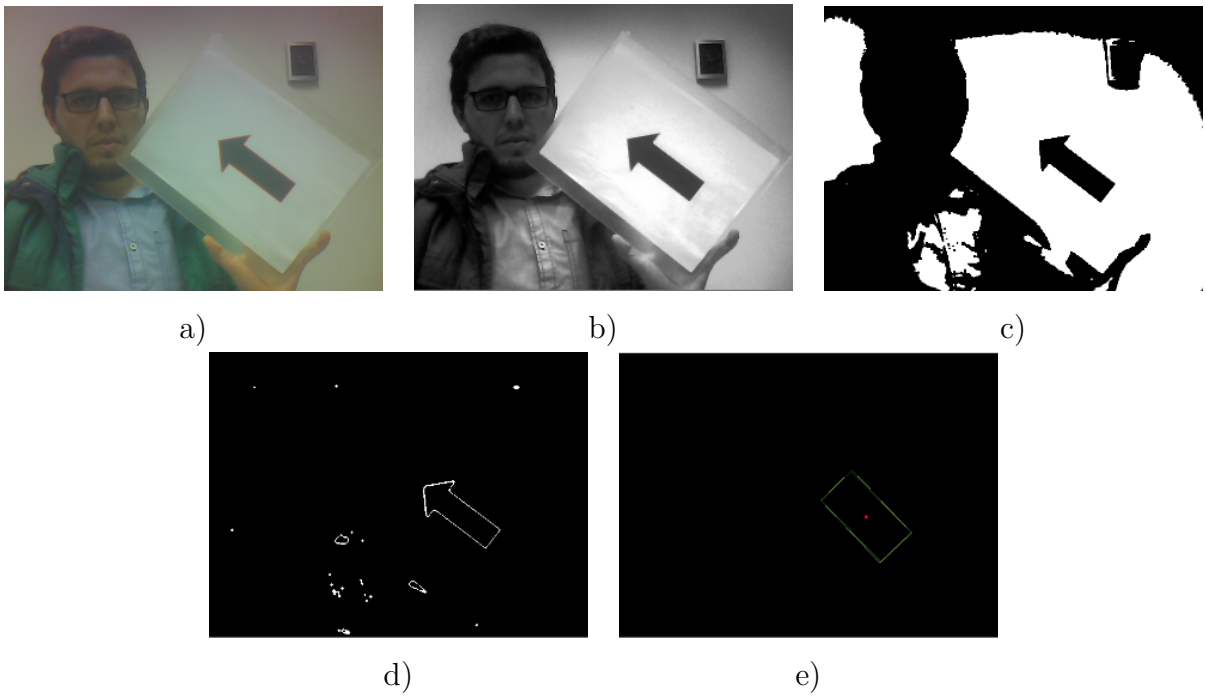


Figure 4.3: Image processing algorithm through the various stages; a) Original color image; b) Grey-scale image; c) Black and White; d) Black blobs contours; e) Target's bonding box.

If the sorting criteria results in one isolated blob, it is assumed that it matches the target and so the center is extracted. This algorithm was implemented inside a ROS node that subscribes the last published frame and publishes the image's target position. Taking advantage of the standard messages that the ROS has by default, it was used a standard *point* message to publish the center position of the target.

4.1.2 Blog Tracking

Instead of finding the target in the hole image on each following frame, it was implemented a tracking algorithm taking advantage of ViSP, in order to optimize the computation efficiency.

Regarding the tracking feature (blob) algorithm, ViSP has several algorithms that may be applied depending on the application, such as: blob tracking, key-point tracking, moving-edges tracking, among others. In the context of this work, it was used the blob tracking algorithm, given the fact that to initialize the tracking it is

only necessary to input the blob's center (that is provided by the target detection algorithm) and the fact that the algorithm is scale invariant (does not depend on the target's distance from the camera variation). Moreover, ViSP refers that the algorithm is robust if noise and specularity is not too strong [20].

Figure 4.4 illustrates one of the tests of the blob tracking algorithm performed in the LAR.

Figure 4.4: ViSP blob tracking.

The implemented algorithm to perform the tracking of the target is based on the ViSP blob tracking algorithm, referred on previous Section 3.3.3 of Chapter 3; it was adapted to initialize from the target position received by the *find target* algorithm within a ROS message containing the coordinates of the target's center position. If the target is lost from the FOV during the PTU's tracking performance, i.e., the tracking system can not maintain the target in the FOV irrespectively of the disturbance made by the humanoid, the algorithm throws an exception; the algorithm re-initializes using the following target position received from the *find target* node. The main cause for losing the target is the velocity of the movements comparing to the camera and the change of luminosity/reflection. During the process, on each iteration the *track arrow* node sends the position of the target to the PTU controller node.

4.1.3 Vision System Evaluation

The tests for the computer vision algorithms (*find arrow* and *tracking*) were based on the foundation to investigate (with a given frame) how long the algorithm takes to get the target's center position and explore its capability to react with the target's

movements (taking into account the employed computer processor). The target movements are related to the actions of the head, producing a displacement of the target on the image.

The experiment was carried out attaching the target on the Fanuc's end-effector and laying the camera in such a way that it is still and directed towards the target's range of movement over the experiments. Along with the first experiment, it was recorded the data from the target position of the "find arrow on image" and the "track arrow" algorithms results on a .csv file taking advantage of the *rosbag* ROS feature.

During the experiments, the Fanuc robot movements were made along with the yy axis in a linear movement with the profile, as illustrated on the following Figures (blue line). It was ensured that the target was in the FOV in the whole range of the movement. Beside the fact that the movement was the same in the three experiments, the velocity for the first experiment was $0,436\,m/s$ corresponding to a 20% of the allowed maximum speed of the robot. The other two were made for a velocity of $1,120\,m/s$ and $1,814\,m/s$. The following charts have two yy axis, one on the left and one on the right. The left side concerns to the position of the Fanuc robot end-effector over the yy axis[2]; the right side relates to the results of the computer vision algorithms. The values of both left and right yy axis are related to the initial position of the experiment.

Figure 4.5 represents the data for the experiment with the maximum speed of the Fanuc robot, $0,436\,m/s$. The Figure shows that the algorithms were capable to correctly track the target. Nevertheless, it should be noted in the chart b), the *find arrow node* data, regarding the target's yy position on the image, presents some variations in the position of the target center. Although the variations are not reflected on the *track arrow* algorithm, they depend on the data provided by the *find arrow node*.

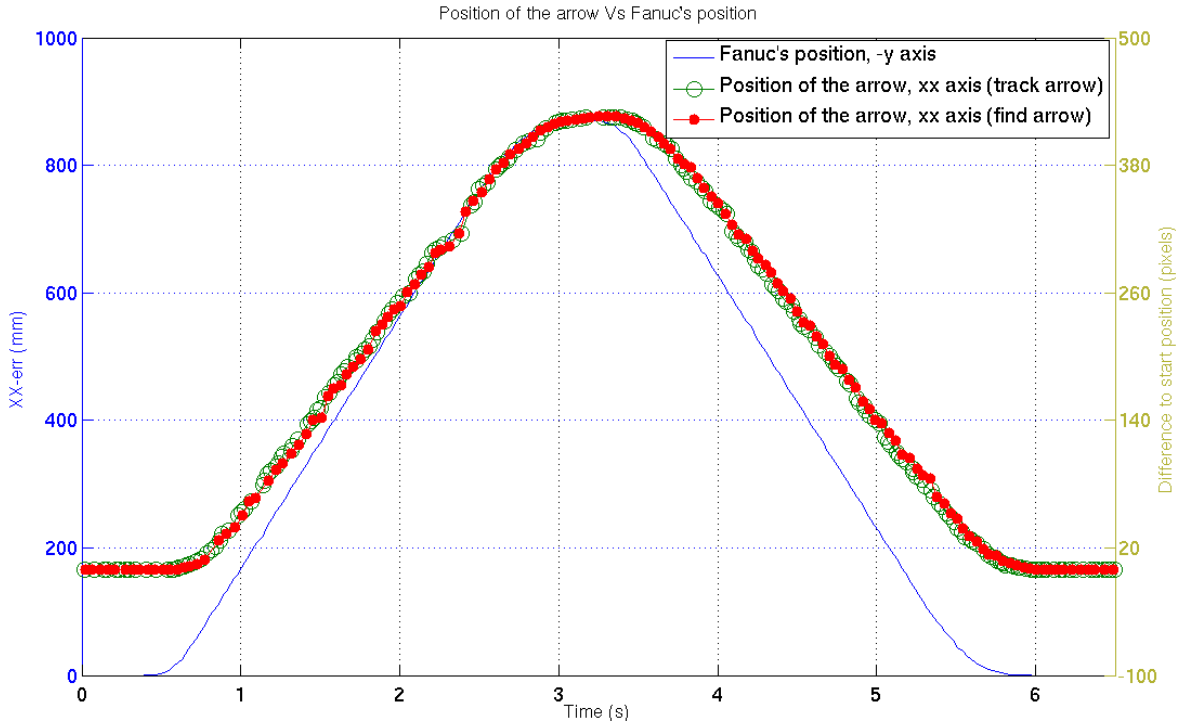
Figure 4.6 represents the data for the experiment with the maximum velocity of the Fanuc robot $1,120\,m/s$. The behavior is similar to the previous experiment and once again it is noticed that the *track arrow* is smoother than the *find arrow*.

Illustrating the experiment with the maximum velocity of the Fanuc robot $1,814\,m/s$, Figure 4.7 shows that at this velocity, the algorithm misses the target in a considerable period of time ($0,4s\,to\,0,8s$). Only when the *find arrow* algorithm finds again the target's position, the *track arrow* restarts tracking the target. It is noteworthy that in the period between $1s\,to\,1,6s$ the *find arrow* misses again the position of the target. However, the *track arrow* continues to track the target. This may be explained by the fact that given such high velocity of the robot, the image captured by the camera becomes blurred, even though the camera is a global shutter one. This situation can

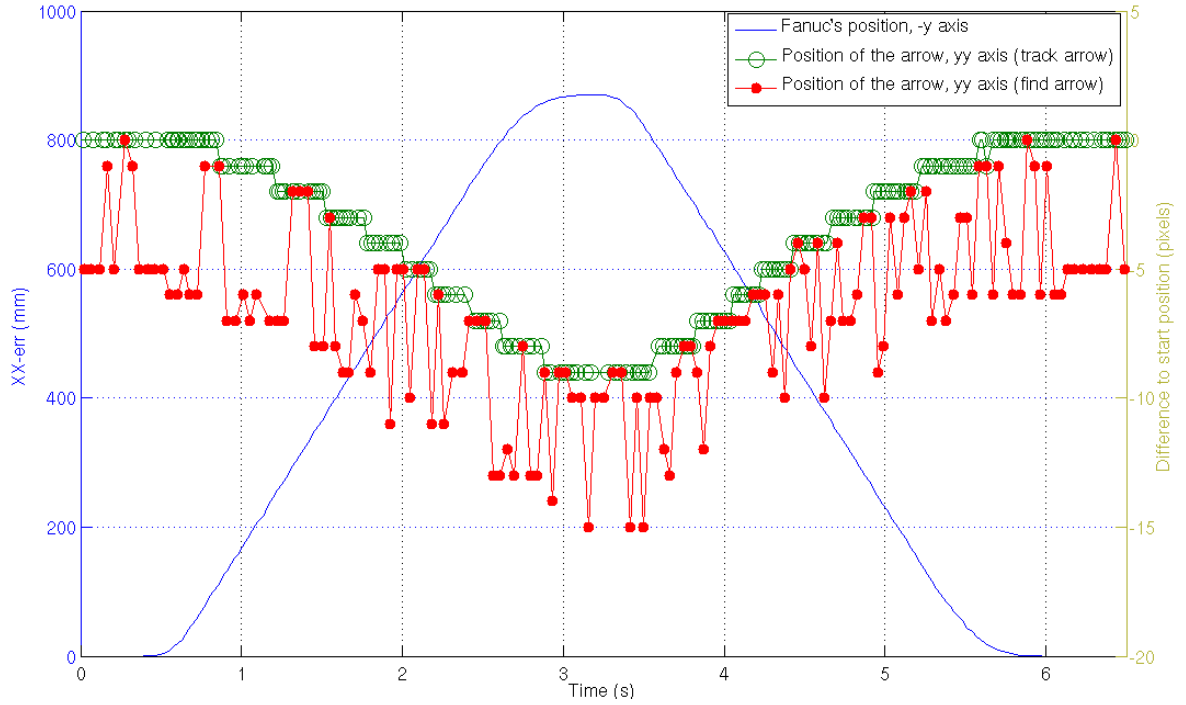
[2]The axis with the most relevant information (along which the movement is more significant).

be circumvented by reducing the exposure time, however, it would require to open the camera lenses diaphragm or increase the ISO sensibility.

Since the *track arrow* algorithm does not take into account the target shape characteristics, even though with a worse image quality, the target blob can still be tracked.

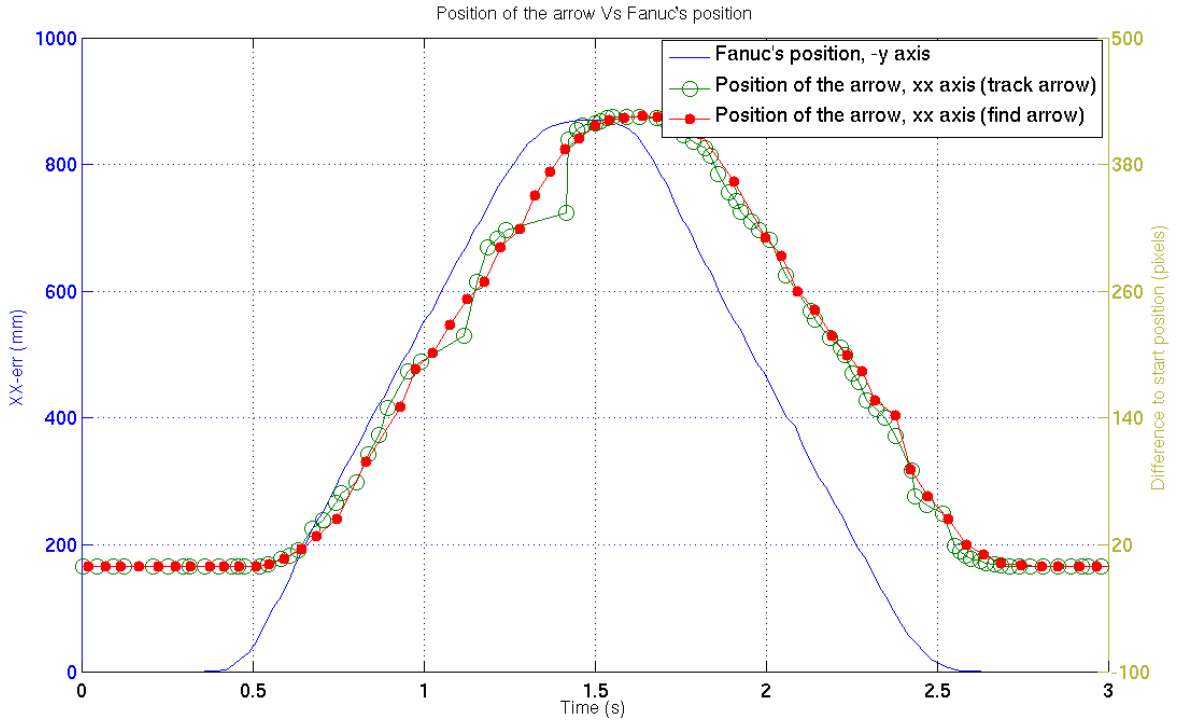


a)

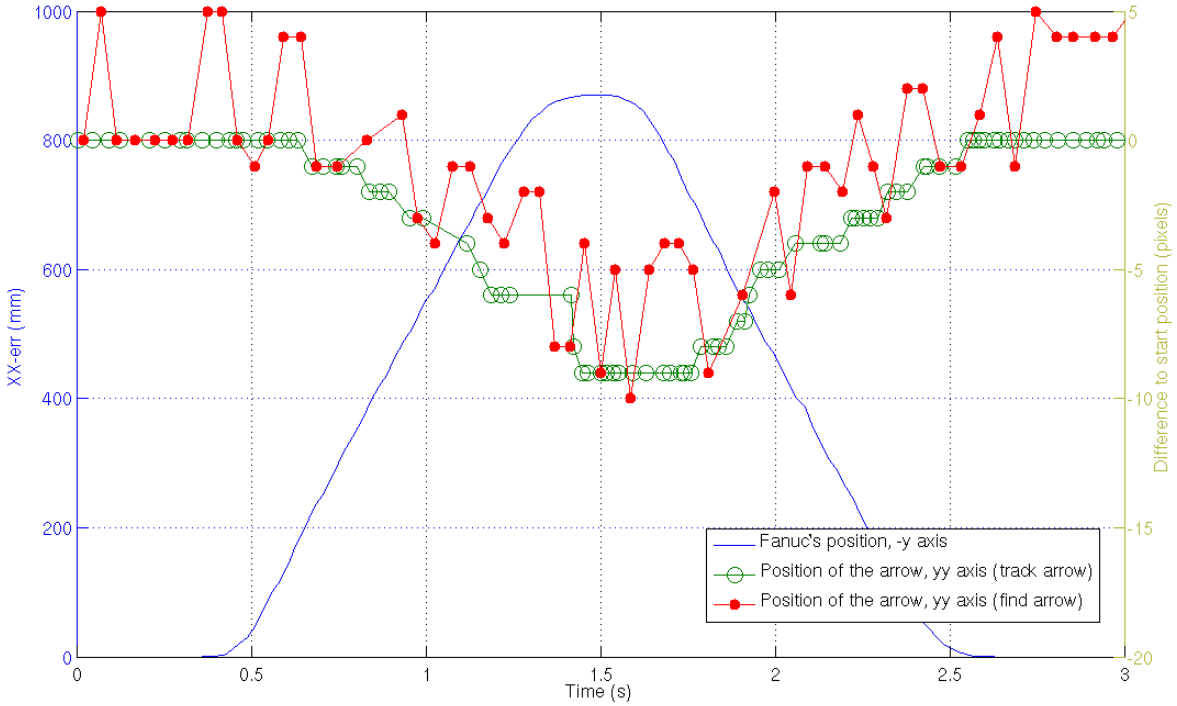


b)

Figure 4.5: Graphic results of the image processing experiments for the Fanuc's velocity maximum velocity of $0,436\,(m/s)$. a) Results for the target x center position on the image. b) Results for the y target center position on the image.

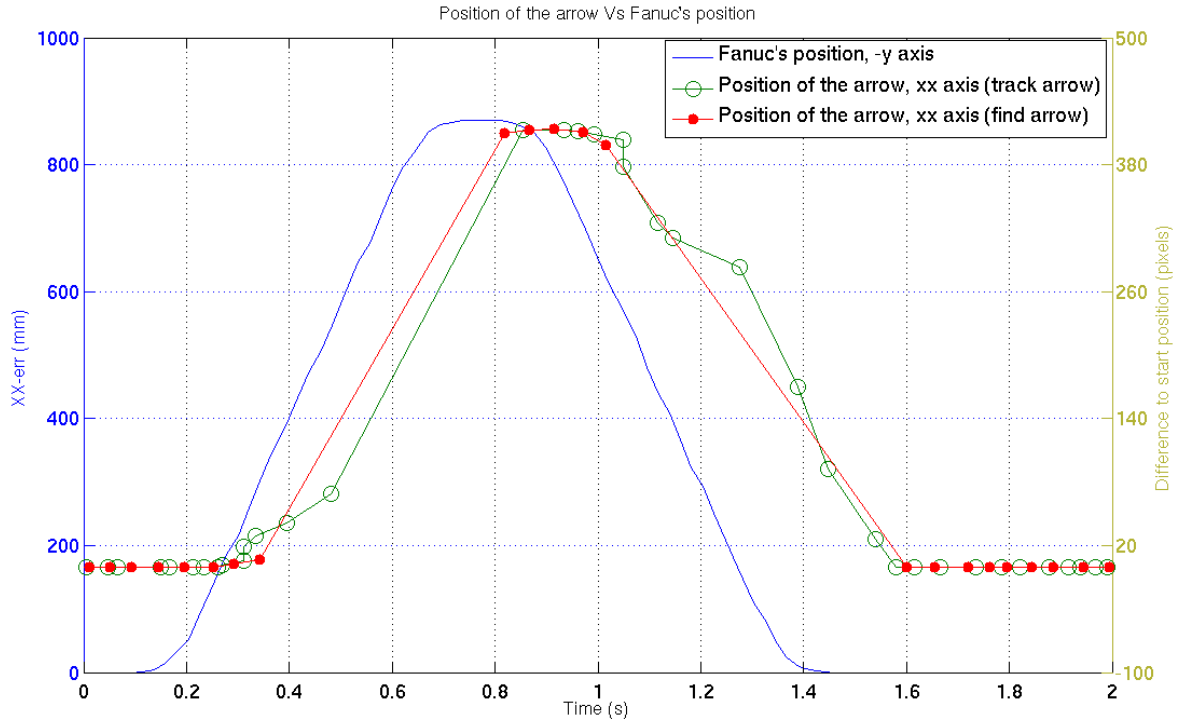


a)

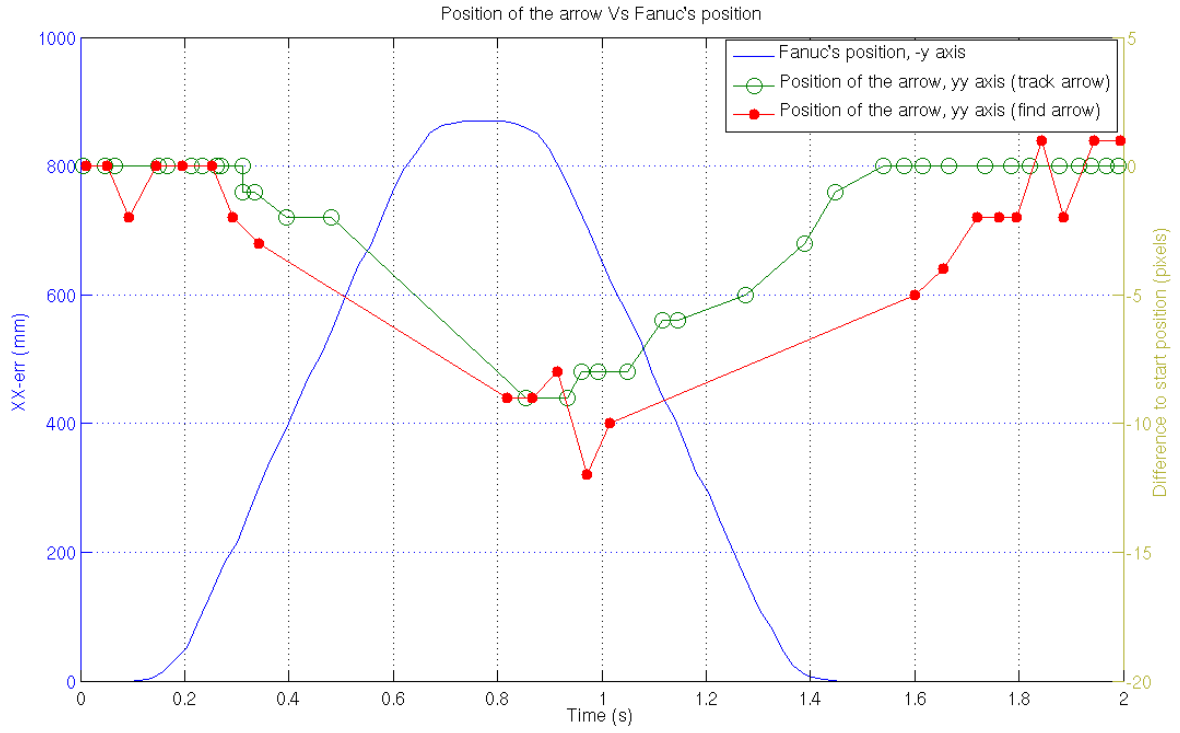


b)

Figure 4.6: Graphics results of the image processing experiments for the Fanuc's velocity maximum velocity of $1,120\,(m/s)$. a) Results for the x target center position on the image, b) Results for the y target center position on the image.



a)



b)

Figure 4.7: Graphics results of the image processing experiments for the Fanuc's velocity maximum velocity of $1,814\,m/s$. a) Results for the target x center position on the image, b) Results for the target y center position on the image.

Figure 4.8 illustrates the time periods between each published message data from the *track arrow node* at the experiment with the Fanuc velocity of $0,436\,m/s$ on a chart. The data shows that the time periods are randomly distributed in the range of $0,05s - 0,01s$ with an average of $0,03s$ $(33\,Hz)$.

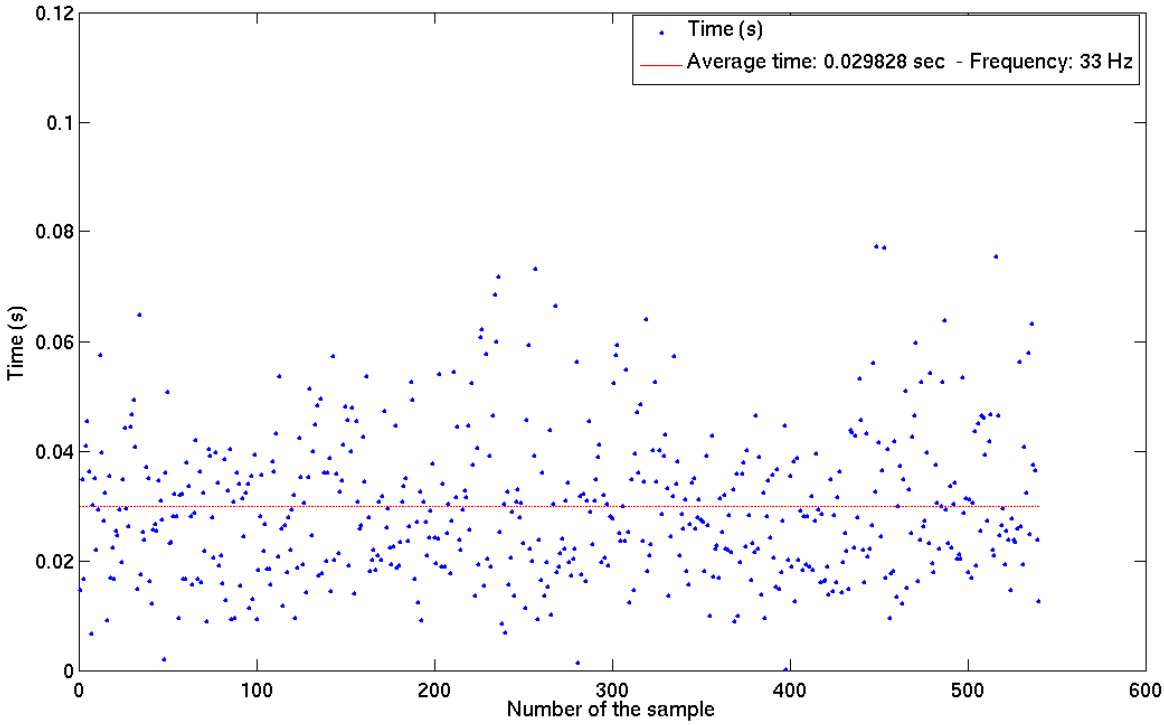


Figure 4.8: Example of the time between target's position values of the "find arrow" ROS node.

The Table 4.1 shows the frequency of the published data by the ROS nodes during the experiments. The data indicates that the average frequency of the *track arrow* node is 64% higher than the *find arrow* node. Moreover, it is verifiable on the illustrated experiments that the frequencies do not present relevant variation between the experiments.

Table 4.1: Average frequency of the published data by the ROS nodes during the experiments.

Fanuc's Velocity	Track arrow	Find arrow	Fanuc data
$0,436\,m/s$ (20%)	$33\,Hz$	$21\,Hz$	$37\,Hz$
$1,120\,m/s$ (50%)	$32\,Hz$	$19\,Hz$	$38\,Hz$
$1,814\,m/s$ (90%)	$30\,Hz$	$18\,Hz$	$37\,Hz$

As expected, the data shows that the performance of the *track arrow* is faster than the *find arrow node*, which proves the advantage to use a tracking algorithm.

Regarding the images from the camera (frames), as expected, only when the active node is the image grabber, it is possible to acquire images at the maximum frame rate,

60*fps*. Nevertheless, along with the activation of the several nodes, the frame rate decreased substantially.

The following chapter describes how the neck's control algorithm was designed and implemented, given the distance between the target's center position in the image to the center of the image.

4.2 Visual Servoing

This section describes the algorithm used for controlling the motion of the PTU in closed loop with respect to visual data. As mentioned before, the vision-based control scheme aims to provide the gaze direction relative to selected target through time. Figure 4.9 illustrates the implemented PTU's control system. The dash-dotted rectangle represents the servomotors inner control system. The visual feedback block is the one described before one the computer vision algorithms (find and track the target on the image, providing its center position feedback).

The servomotors have two different parameters to control them, which are the position and the velocity. Therefore, to implement a control algorithm with visual feedback, there are two possibilities that need to be validated for finding the one that is best suited in the specific context of this work.

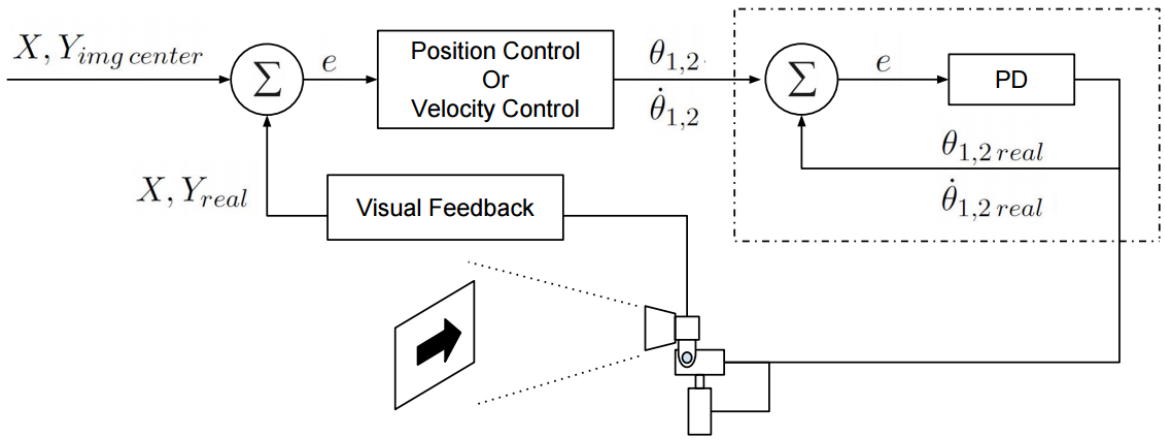


Figure 4.9: PHUA PTU's Control system.

The control by velocity is an approach that changes the velocity $\dot{\theta}$ of the servomotors according to the difference between the target's center and the center of the image Δ_{error}. The SP position is defined to the extremes of the servomotors range, depending on the direction of the movement.

$$\dot{\theta}_i = K\Delta_{error} \tag{4.1}$$

Symmetrically to the velocity control, the control by position method varies the SP position θ_i based on the difference between the target's center position and the center of the image $K\Delta_{error}$ with the maximum velocity.

$$\theta_i = \theta_{i-1} + K\triangle_{error} \tag{4.2}$$

As a first approximation, we decouple the pan and tilt joint motions such that they are directly related to the displacement of the target relative to the image's center position in each direction. Independently of the control mode, a proportional controller is used, meaning that the position (or the velocity) of the servomotors is proportional to the distance of the SP to the center of the image. The implementation of the vision-based PTU controller assumes the existence of a variable gain: in an initial phase (detection), the algorithm uses a small gain value such that it smoothly position the PTU until it faces the target point (arrow centroid point) in the middle of the FOV. After reaching that point (the SP), a higher proportional gain is introduced. This is the one that should be tested in order to find the best to enable the PTU to track the target as fast as possible without entering under instability.

4.3 Ego-motion Estimation

The ability to perceive the three-dimensional motion in relation to the environment is crucial for autonomous robots acting in a dynamically changing world. Ego-motion estimation for humanoid robots can help in stabilization and navigation tasks, for example, by modulating the gait characteristics such as speed and stride length. In this work, it is investigated how fixation on a static visual cue can be used to estimate some motion parameters useful for a humanoid robot, such as inferring the speed and direction in a scene. Fixation over time changes the input in a controlled way and from this change additional information is derived.

This section derives the equations that translate the information from the PTU into motion parameters. In particular, there are two parameters that must be inputed, which are the distance of the target to the PTU (future stereo vision system) and the angles of the PTU that are read from the servomotors low-level controllers. In what follows, it is assumed that the PTU is moving in straight line. The calculations are performed in two situations, when the PTU is moving forward and sideways.

Figure 4.10 represents a situation when the PTU is moving in one direction; in this particular case, from point A to point B.

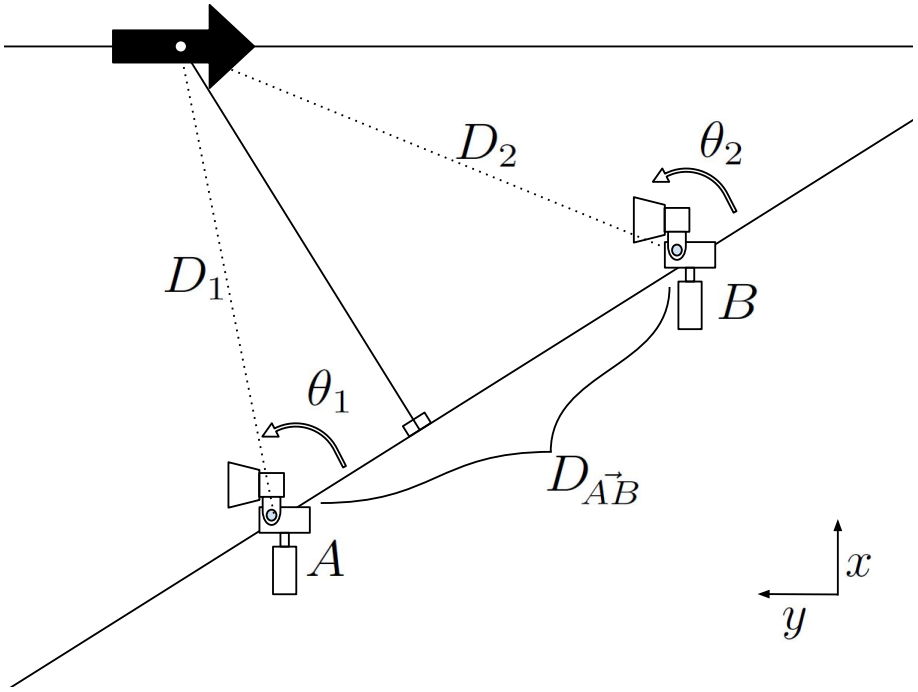


Figure 4.10: Schematic of the ego-motion calculation for the *pan* component of the PTU.

The angles θ_1 and θ_2 correspond to *pan* angle between the vector that starts from the PTU towards the target D_1 and the direction of the movement; those angles correspond to the moment A and B respectively. The distance from point A to point B is $D_{\vec{AB}}$. Therefore, the velocity of the PTU (and hence the PHUA) is $v_{_{PTU}} = \frac{D_{\vec{AB}}}{\Delta t}$ regarding the xy-plane. The calculus of $D_{\vec{AB}}$ relies upon the way that the PTU is moving, i.e., forward or sideways. In case of the sideways movement, (such as Figure 4.10 suggests) $D_{\vec{AB}}$ is as follows:

$$D_{\vec{AB}} = D_1 \times \cos(\theta_1) + D_2 \times \cos(180^{\circ} - \theta_2) \tag{4.3}$$

Regarding the forward movement, the equation to obtain $D_{\vec{AB}}$ is:

$$D_{\vec{AB}} = -D_1 \times \sin(\theta_1) + D_2 \times \sin(\theta_2) \tag{4.4}$$

The calculation of the height h of the PTU in relation to the position of the target is made with a simple trigonometric equation such as Figure 4.11 suggests.

The height h is given by:

$$h = D_1 \times \sin(\theta_1) \tag{4.5}$$

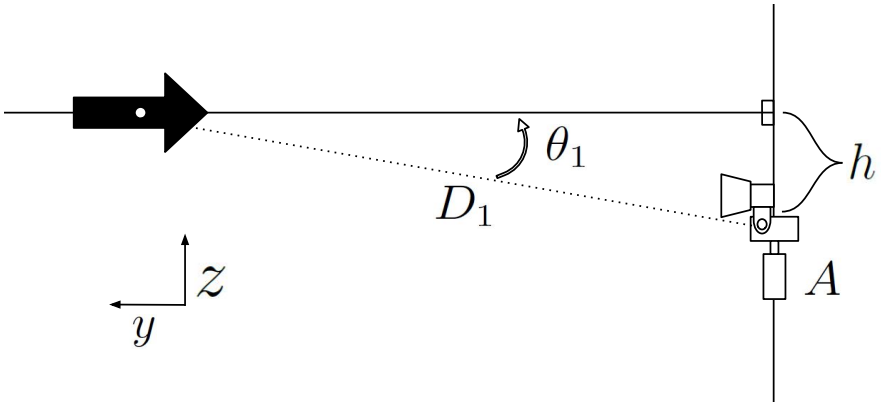


Figure 4.11: Schematic of the ego-motion calculation for the *pan* component of the PTU.

Therefore, the velocity of height change is $v_h = \frac{h(t+\triangle t)-h(t)}{\triangle t}$.

It is noteworthy that the distance D_1 is the 3D distance from the target to the PTU, regarding the calculation of the height h.

The previous equations allow to estimate the position of the PTU in relation to a target, with a given angle of the PTU and the information of the distance of the target. As a result, the information obtained will help the PHUA to navigate in an ambient with this VO system.

The following Figures 4.12 and 4.13 illustrate two experiments made to test the previously described ego-motion equations. The first experiment was made with a simple movement of the Fanuc robot along with the yy axis, as Figure 4.12 suggests. The second experiment was made along with the zz axis represented in the Figure 4.13. In both experiments the ego-motion equations are validated. There is a small difference between the estimation and the real curves coming from rounding in the trigonometric functions. The data of the angles of the PTU are provided by a MATLAB simulator. The data analyses of the ego-motion estimation with real data performed with the PTU in the experiments are presented in the following chapter.

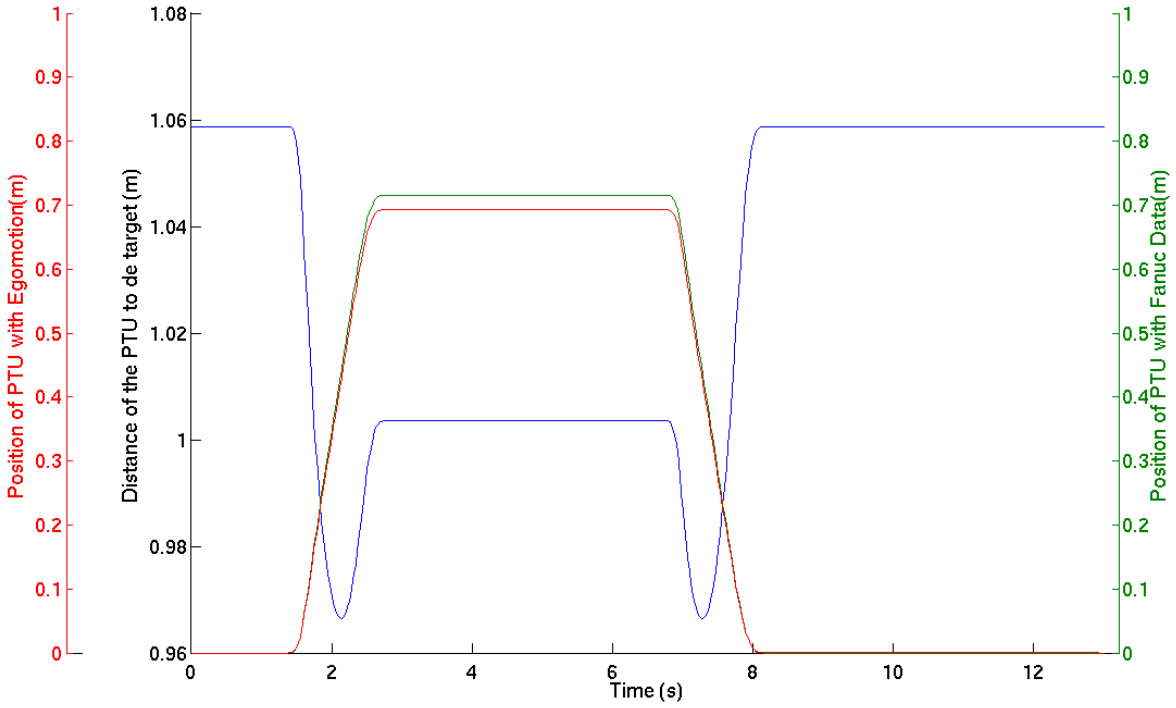


Figure 4.12: Distance of the PTU to the target in the XY plane (blue line), position of the PTU with Fanuc data (green line) and ego-motion estimation of the position of the PTU (red line)

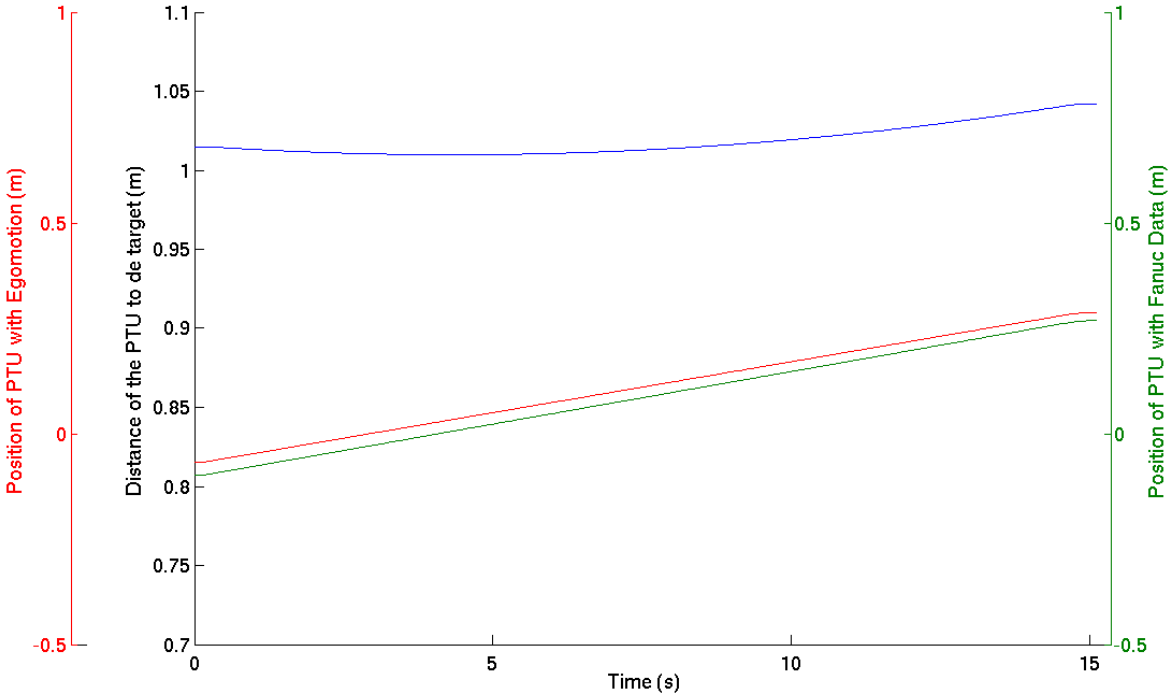


Figure 4.13: Distance of the PTU to the target (blue line), position of the PTU with Fanuc data (green line) and ego-motion estimation of the PTU's height in relation to the target(red line)

The following chapter 5 describes the results obtained in the estimation of the ego-motion.

4.4 PTU Simulator

With the aim of simulating the behavior of the PTU's control system once applied an external movement on its base reference, it was developed an MATLAB script simulator to analyze and predict the response behavior of the PTU. This simulator integrates the data from the movement of the PTU's reference and the PTU's joint positions.

Based on the external movement position data (position of the PTU's reference in relation with the global position), the PTU's forward kinematics and the position of the visual target in relation to global reference, the simulator determines the ideal SP angles that the control system needs to send to the PTU's servomotors in order to correctly track the visual target.

The simulator also has the ability to compute a PTU's reference trajectory given an initial and final point. It generates the position and velocity profile or, in case of the position over time is available on an external data file, it is possible to read it and extract the trajectory data. This aspect will be very important in the experimental setup described in Chapter 5.

Another future of this script is the ability to estimate the robot's ego-motion using the VO equations presented on Section 4.3. Following subsections explains how the equations to build the simulator were calculated.

4.4.1 PTU's Forward Kinematics

After the structure was made, in order to make a kinematic characterization of the neck, it was calculated the neck's forward kinematics following the Denavit–Hartenberg algorithm (1955). This may be used to plan or study the movements or even in the future for further improvements and analysis [70].

Analyzing the Figure 4.14 and following the Denavit-Hartenberg rules, it was obtained the Table 4.2.

Table 4.2: Direct Kinematics of neck.

Link	θ	α	l	d
1	θ_1	90^o	0	l_1
2	$\theta_2 + 90^o$	0	l_2	0
3	-90^o	0	l_3	0

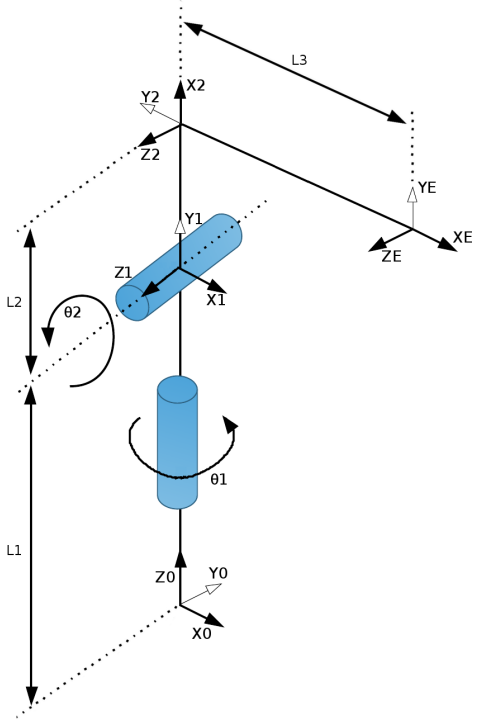


Figure 4.14: Neck's kinematic schematic.

Given the Denavit-Hartenberg table, it was applied the equation 4.7 that represents the transformation matrix for each link.

$$A_i = Rot(z, \theta_i) \times Trans(l_i, 0, d_i) \times Rot(x, \alpha_i) \tag{4.6}$$

$$A_i = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & l_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & l_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.7}$$

Applying the matrix 4.7 for each link T of the neck, the correspondent matrices are given in 4.8.

$$A_1 = \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A_2 = \begin{bmatrix} -s_2 & -c_2 & 0 & -l_2 s_2 \\ c_2 & -s_2 & 0 & l_2 c_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A_3 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -l_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.8}$$

Multiplying all matrices, it is obtained the matrix 4.10 that relates the link 0 with the end- effector E, (^0T_E), which in this case corresponds to the focal point of the camera.

$$^0T_E = {}^0T_1 \times {}^1T_2 \times {}^2T_E \tag{4.9}$$

$$^0T_E = A_1A_2A_3 = \begin{bmatrix} \begin{bmatrix} c_1c_2 \\ s_1c_2 \\ s_2 \end{bmatrix} & \begin{bmatrix} -c_1s_2 \\ -s_1s_2 \\ c_2 \end{bmatrix} & \begin{bmatrix} s_1 \\ -c_1 \\ 0 \end{bmatrix} & \begin{bmatrix} l_3c_1c_2 - l_2s_2c_1 \\ l_3s_1c_2 - l_2s_2s_1 \\ l_3s_2 + l_2c_2 + l_1 \end{bmatrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.10}$$

With this equation, it is now possible to extract the camera's position (humanoid eyes) and the direction for a given angle position of the neck's PTU. For instance, the end-effector position of the PTU p_x, p_y and p_z is therefore:

$$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} l_3c_1c_2 - l_2s_2c_1 \\ l_3s_1c_2 - l_2s_2s_1 \\ l_3s_2 + l_2c_2 + l_1 \end{bmatrix} \tag{4.11}$$

Note that p_z only depends on the θ_2 angle and the $\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$ depends from one to another or in other words, they are not independent form each other.

4.4.2 PTU's Inverse Kinematics

As mentioned before in Section 4.4.1, the forward kinematics are the direct result of the joint angles computation of a given robot system in order to calculate the end-effector's position. Therefore, the forward kinematics are the equations that, given a set of joint positions, obtain the end-effector position of a robotic system. In contrast with forward kinematics, the inverse kinematics determines the joint angles of a given robotic end-effector position. This equations assume a big importance given the fact that in practice, the PTU's controller needs to calculate the servomotors angles in each iteration in order to be able to track the target.

The PTU's inverse kinematics equations were calculated to equip the MATLAB simulator with the capability to compute the joint angles given a certain position of the PTU's reference, as well as the target position. Note that in this context, the inverse kinematics are calculated to get the PTU's joint positions given a certain position of the target and the global reference of the PTU.

The equation 4.12 shows the PTU's forward kinematics equation of the ${}^0T_E$ obtained in previous Chapter 3 [3].

$$
{}^0T_E = \begin{bmatrix} \begin{bmatrix} c_1c_2 \\ s_1c_2 \\ s_2 \end{bmatrix} & \begin{bmatrix} -c_1s_2 \\ -s_1s_2 \\ c_2 \end{bmatrix} & \begin{bmatrix} s_1 \\ -c_1 \\ 0 \end{bmatrix} & \begin{bmatrix} l_3c_1c_2 - l_2s_2c_1 \\ l_3s_1c_2 - l_2s_2s_1 \\ l_3s_2 + l_2c_2 + l_1 \end{bmatrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.12}
$$

Throughout the previous equation 4.12, interpreting the proposed PTU's structure illustrated on Figure 4.14, the position of the end-effector of the PTU[4] on the ZZ axis does not depend on the θ_1 ($P_Z = [l3s2 + l2c2 + l1]$), as expected. Considering the simplified PTU's schematic from the top view presented on Figure 4.15 and the previous forward kinematics equation 4.12, the θ_1 is calculated by the equation 4.13.

$$
\theta_1 = \arctan\left(\frac{PX}{PY}\right) \tag{4.13}
$$

The θ_1 drives the PTU towards the vertical plan that contains the target's reference coordinates.

After the calculus of joint θ_1, the PTU and the target can be represented on a plane such as Figure 4.16 portrays.

By analyzing Figure 4.16, when the PTU is aligned with the target T, the link 2, the end-effector E and the target T belong to the same beeline. To obtain the equation of this beeline ($\overrightarrow{r}$), it is necessary to know two parameters; (1) the position vector ($\overrightarrow{a}$) of one point that is part of the beeline $\overrightarrow{r}$ and (2) the vector $\left(\overrightarrow{b}\right)$ that gives the direction of $\overrightarrow{r}$. As a result, any point on the vector $\overrightarrow{r}$ is given by the equation 4.14.

$$
\overrightarrow{r} = \overrightarrow{a} + t \cdot \overrightarrow{b} \tag{4.14}
$$

Applying the equation 4.14 in the context of the problem, considering the joint 2 (${}^0T_2$), $\overrightarrow{a}$ is the position vector: $\overrightarrow{a} = \begin{bmatrix} -l_2s_2c_1 \\ -l_2s_2s_1 \\ l_2c_2 + l_1 \end{bmatrix}$. Regarding the vector $\overrightarrow{b}$, it has the same direction as X_E, therefore, from the equation 4.12 comes: $\overrightarrow{b} = \begin{bmatrix} c_1c_2 \\ s_1c_2 \\ s_2 \end{bmatrix}$. Applying the values of $\overrightarrow{a}$ and $\overrightarrow{b}$ on the equation 4.14, it is obtained the equation that relates any point of the vector $\overrightarrow{r}$ with PTU's reference (0):

[3] c_1 suits for $\cos\theta_1$ and so forth.

[4] Camera's Focal Point.

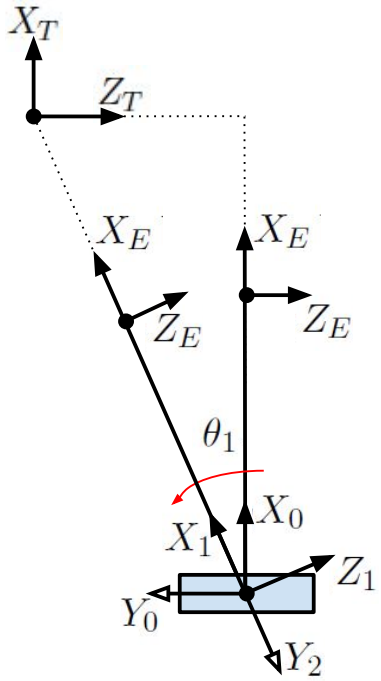


Figure 4.15: Schematic used to calculate the PTU's θ_1 (view from the top).

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} -l_2 s_2 c_1 \\ -l_2 s_2 s_1 \\ l_2 c_2 + l_1 \end{bmatrix} + t \cdot \begin{bmatrix} c_1 c_2 \\ s_1 c_2 \\ s_2 \end{bmatrix} \tag{4.15}$$

If the previous values $\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$ are the position of the target $T\left(\begin{bmatrix} X_T \\ Y_T \\ Z_T \end{bmatrix}\right)$ and the PTU is aligned with T, there is a value t that suits the condition of the next equation 4.16.

$$\begin{bmatrix} X_T \\ Y_T \\ Z_T \end{bmatrix} = \begin{bmatrix} -l_2 s_2 c_1 \\ -l_2 s_2 s_1 \\ l_2 c_2 + l_1 \end{bmatrix} + t \cdot \begin{bmatrix} c_1 c_2 \\ s_1 c_2 \\ s_2 \end{bmatrix} \tag{4.16}$$

$$=> t = \frac{X_T + l_2 s_2 c_1}{c_1 c_2} = \frac{Y_T + l_2 s_2 s_1}{s_1 c_2} = \frac{Z_T - l_2 c_2 - l_1}{s_2} \tag{4.17}$$

The solution for the previous equation 4.17 can be achieved throughout the manipulation of itself, to obtain an equation form such as the 4.18. This equation has a known solution that is demonstrated on 4.19.

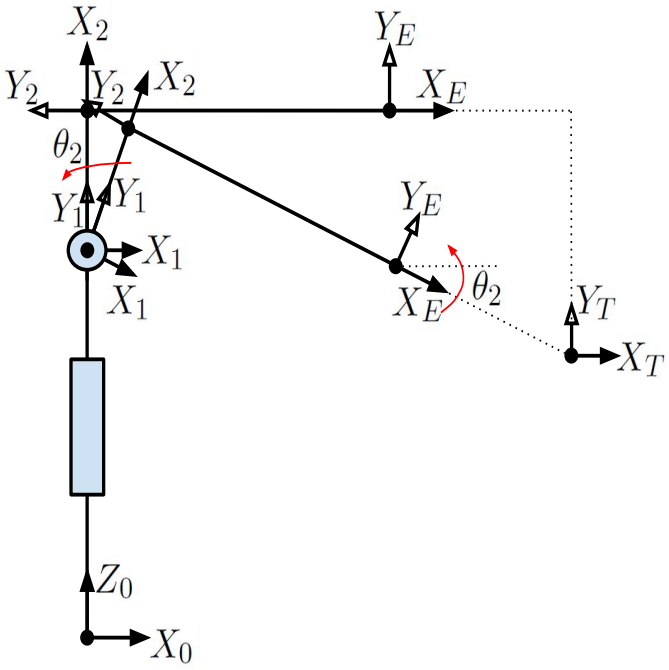


Figure 4.16: Schematic of the plane that contains the PTU and the target.

$$K_1 \sin(\theta) + K_2 \cos(\theta) = K_3 \tag{4.18}$$

$$\theta = \arctan\left(\frac{K_1}{K_2}\right) \pm \arctan\left(\frac{\sqrt{K_1^2 + K_2^2 + K_3^2}}{K_3}\right) \tag{4.19}$$

Therefore, manipulating the equation 4.17 in order to have the form of 4.18, two following solutions are accomplished:

1.

$$\frac{X_T + l_2 s_2 c_1}{c_1 c_2} = \frac{Z_T - l_2 c_2 - l_1}{s_2} <=> X_T s_2 + l_2 c_1 s_2^2 = Z_T c_1 c_2 - l_2 c_1 c_2^2 - l_1 c_1 c_2 <=>$$

$$<=> \underbrace{X_T}_{K_1} \sin(\theta_2) + \underbrace{(l_1 - Z_T) c_1}_{K_2} \cos(\theta_2) = \underbrace{-l_2 c_1}_{K_3}$$

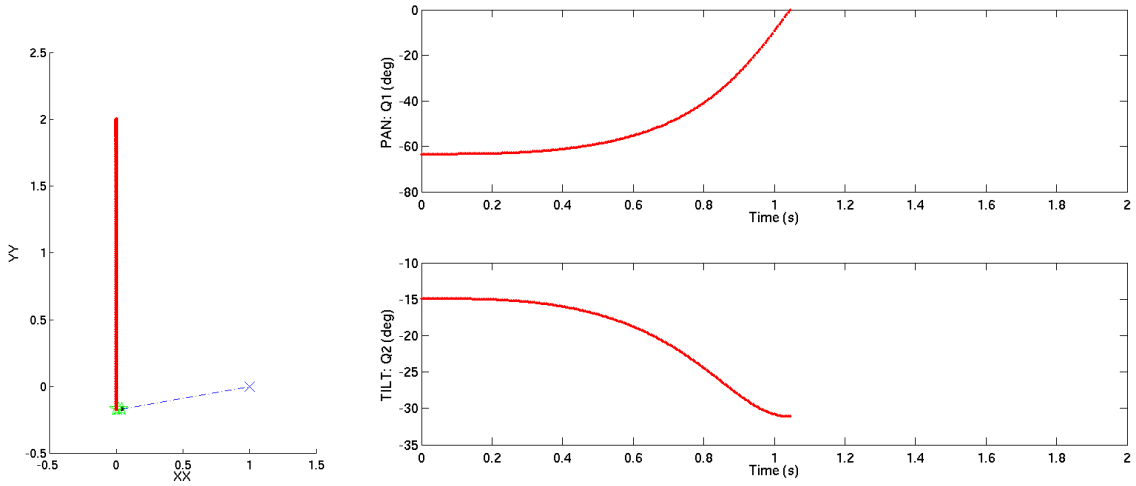
2.

$$\frac{Y_T + l_2 s_2 s_1}{s_1 c_2} = \frac{Z_T - l_2 c_2 - l_1}{s_2} <=> Y_T s_2 + l_2 s_1 s_2^2 = Z_T s_1 c_2 - l_2 s_1 c_2^2 - l_1 s_1 c_2 <=>$$

$$<=> \underbrace{Y_T}_{K_1} \sin(\theta_2) + \underbrace{(l_1 - Z_T) s_1}_{K_2} \cos(\theta_2) = \underbrace{-l_2 s_1}_{K_3}$$

This two equations were integrated on the MATLAB simulator to compute the inverse kinematics. Note that there are two particular cases on the previous equations which are not valid: on the first (1.) when $\theta_2 = \pm 90^o$ and on the second (2.) when $\theta_2 = 0^o \smile \theta_2 = 180^o$. The implemented solution chooses the solution (1.) or (2.) according to θ_1 and the signal of the solution 4.19 consonant with the signal of X_T and Y_T respectively.

Taking advantage of the computation of the forward and inverse kinematics, Figure 4.17 illustrates a screenshot at a given moment of a linear movement in one simulation example: on a) it is illustrated the trajectory in a red dotted line, the blue 'x' represents the target and the green part is the PTU, on b) it is represented the angles of the PTU (*pan* and *tilt*) over time.



a) b)

Figure 4.17: PTU movement MATLAB simulator. a) Illustration of a PTU's linear movement on the Y axis in relation with the global reference. b) Pan and *tilt* angles over time during the movement a).

Through the developed simulator, it is now possible to compute and simulate a whole range of situations with its trajectory planning feature and predict the behavior of the PTU off-line, as well as extract valuable data for other applications of an PTU without being in the PHUA's context.

4.5 Final Remarks

At this point, the humanoid can search the target on its FOV similarity to a human eye, and track it aided by its PTU controller, alike a human's neck.

With the tracking target algorithms and the implemented PTU's control system, it is possible to make a setup to test the developed work and analyze the robustness of this approach, coming out conclusions of the validity to implement visual feedback into the humanoid's navigation system. Furthermore, regarding the image processing algorithms, they can be used in different project circumstances or contexts given its modularity, requiring only the ROS software platform. For instance, the tracking algorithm that uses the ViSP library is able to track any blob from a given point on the image. Although it is not implemented, the algorithm is easily modifiable to extract other target parameters. It has also become available a *roslaunch* for the camera that can be used later in different contexts.

The developed MATLAB simulator script revels a huge potential given its modularity and the fact that it is possible to simulate a huge number of situations both with the tracking control or the visual odometry estimation of the robot's ego-motion. Moreover, it is not restricted to only the PHUA.

Chapter 5 describes the extracted results from the simulator with the data from the conducted experiments to test all the system.

Chapter 5

Experimental Results and Discussion

This chapter describes the conducted experiments with the purpose of validating the reformulated mechanical structure of PHUA's neck, the tracking algorithms and the validity of the ego-motion estimation. Section 5.1 describes how the experimental setup was structured to develop the experiments and reports the conducted tests and the experimental results. Section 5.2 analyses the results, making use of the developed simulator. Section 5.3 reports the results obtained by applying the ego-motion equations to the experiments and final section 5.4 discusses about the whole work of this dissertation.

5.1 Experimental Results

Based on the previous described experimental setup, this section reports the experiments and the results with the aim to characterize and parameterize the developed system.

The tests of the developed system were divided in two parts. In the first part (1), it was built a setup to evaluate the image processing and the visual tracking algorithms described in the previous chapter, Subsection 4.1.3. After characterizing the performance of the image related algorithms, it was made a second setup (2) to examine the performance of the developed work about the PTU's tracking system. The second experiment was carried out with the target fixed on a table and the PTU grabbed on FANUC, as Figure 5.1 represents.

The performed experiments were conducted in order to verify the PTU's ability to track the target with a given movement of the Fanuc robot. Similarly to the first experiment, it was also used the same computer and the Fanuc robot to produce the

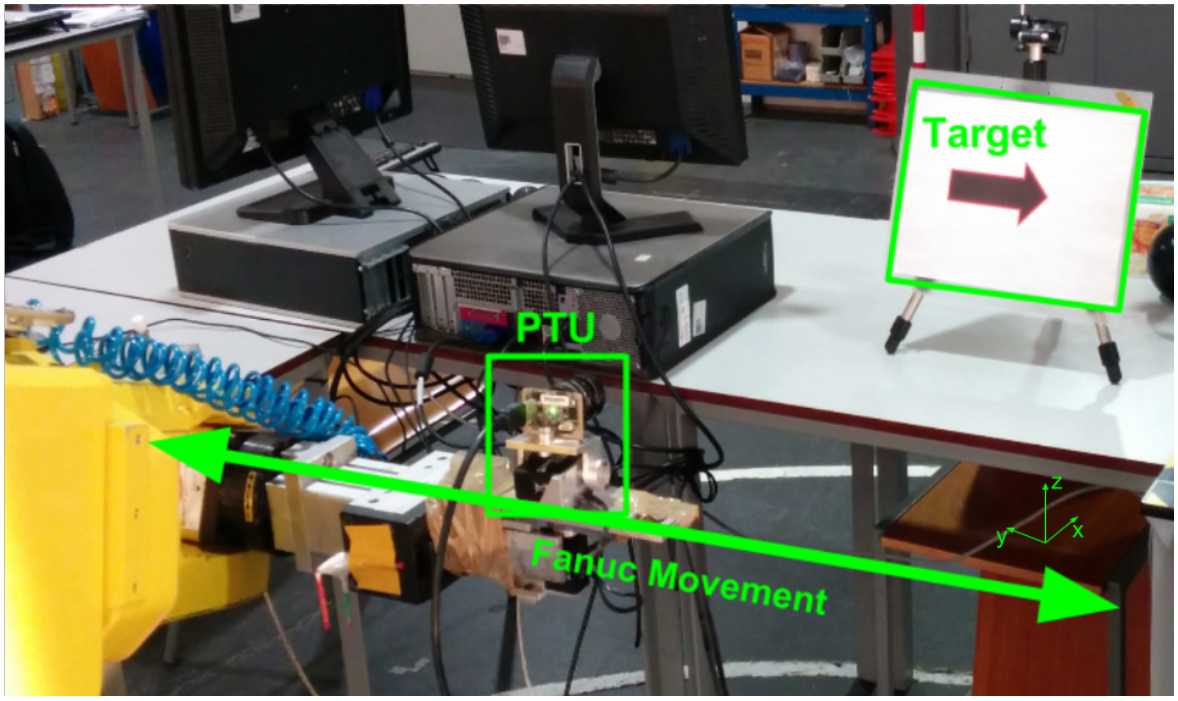


Figure 5.1: Schematic of the experimental setup.

movements, but in this case, it was the PTU that was placed in the robot's end-effector to simulate the humanoid robot. In this experiment, in addiction to the previous recorded data parameters, it was also recorded the data from the PTU position given by its two servomotors.

Next Subsection 5.1.1 addresses the employed metrics to compare and characterize the results of the experiments. It is presented how the experiments were performed, as well as the movements or the parameters of each experiment.

5.1.1 Metrics

Regarding the metrics to analyze and compare the data of the experiments, it was employed the Mean Squared Error (MSE) because it penalizes the major differences (on this context they are considered errors) since the differences are squared. The equation 5.1 was applied to calculate the MSE. The n is the number of values of the experience, i corresponds to the iteration and $\hat{Y}$,Y address the values of the concerned i^{th} iteration.

$$MSE = \frac{\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2}{n} \tag{5.1}$$

An additional method that was applied to measure the performance with regard to tracking control evaluation, was the time of stabilization, which suits for the time

required for the tracking controller stabilize right after the movement of the Fanuc robot stops; the less time, the better.

Concerning the data analysis, it was developed a MATLAB script to extract the data from the *.csv* data files and build charts that display the results, enabling to compare and draw conclusions for all experiments.

Following Subsection 5.1.2 present the experimental results for the developed tracking system.

5.1.2 Tracking Control Evaluation

Based on the experimental setup and the computer vision limits obtained in the previous chapter, this subsection describes the experiments and the results achieved with the final setup all together, including the tracking control algorithms for the PTU's servomotors.

The main objective on testing the system is to find the best algorithm and the optimum parameters to track the target. There are four major parameters involved, which are the velocity and type of movement of the Fanuc robot, the position or velocity tracking control approach as well as the proportional constant K_p for both approaches.

With regard to the execution of external controlled movements on the PTU's reference made by Fanuc robot, in order to test the PTU's tracking capability facing such perturbations, the following three movements for the Fanuc robot to perform were selected:

- Movement A, illustrated on Figure 5.5 (blue line), this is a linear movement in the yy axis (with regard to the global reference and the Fanuc robot); this movement pauses a few seconds before changing direction;
- Movement B, illustrated for instance on Figure 5.3 (blue line), this is also a linear movement along with the yy axis (with regard to the global reference); this movement differs from the previous one in not having any intermediate stops;
- Movement C, this movement is represented on Figure 5.2 with all its positional and rotational components (visually, the movement makes a route similar to three edges of a cube and a rotation between each link[1]).

The objective of this movements are to simulate potential motions of PTU when attached to the humanoid robot; the movements were made individually isolated on each component of the movement at any given time (e.g. from $62s$ to $78s$ in Figure 5.2, solely the zz component is activated), allowing to singly evaluate the tracking control performance. In the case of the movement C, the Figures that contain this movement

[1] Corners

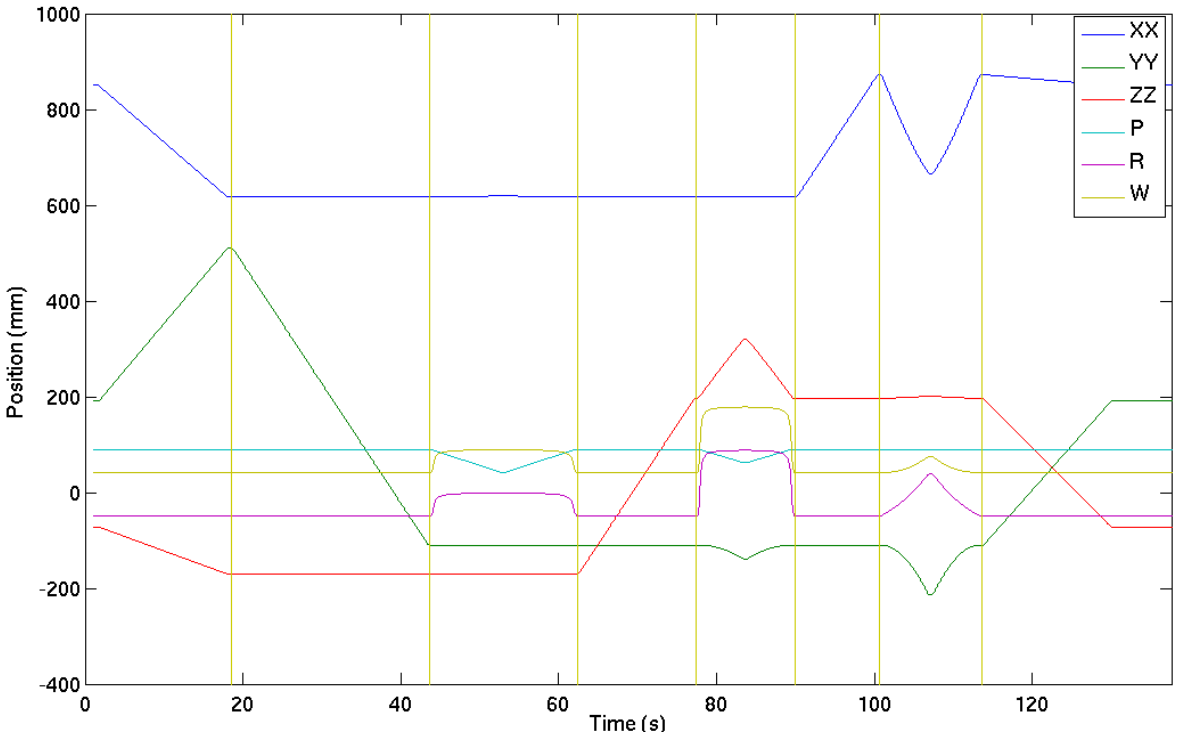


Figure 5.2: Movement C profile of the Fanuc robot end-effector over time.

have vertical yellow lines separating each segment of the movement in order to assist the visualization of the reader (see Figure 5.2).

The movement A aims to evaluate the system when it is faced with a sudden stop (SP), in contrast, the movement B analyses its capability to track, given its uniform movement. The objective of the movement C is to test all possible types of movements with linear and rotational movements; this experiment was made when it was obtained the best parameters for both position and velocity tracking controllers.

As previously stated in Chapter 4, before starting the movement of the robot Fanuc, it was ensured in all experiences that the PTU was staring at the target.

Table 5.1 shows the obtained results for the experiments with the velocity tracking control activated. The displayed experiments suits for the best parameters of K_p for each tested Fanuc robot velocity.

Table 5.1: Results of the experiments for the velocity tracking control.

Experiment	1)	2)	3)	4)
Fanuc's movement profile	Movement A		Movement B	Movement C
Fanuc velocity (m/s)	$0,46$	$1,05$	$0,55$	–
Test time (s)	$8,6$	$6,6$	$8,5$	$128,5$
K_p	$0,1$	$0,05$	$0,1$	$0,1$
MSE Pan	24 460	40 005	39 191	3 626
MSE Tilt	$18,2$	$48,4$	$47,4$	2 722
Stabilization time (s)	$8,2$	$3,9$	$2,4$	$1,1$

The parameter 'Fanuc velocity' relates to the maximum velocity performed on each experiment by the Fanuc robot. The values of the MSE are associated with the measurement of the position of the target (Y_i) to the center of the image $\left(\hat{Y}_i\right)$ in pixels; therefore, the smaller MSE is, the better. If the error is consistently close to zero, denotes that independently of the external perturbations made on the PTU's base reference, the tracking system is actively adjusting the position of the camera to maintain the target in the center of the FOV.

Still in Table 5.1, experiment 2) suggest when the K_p is smaller, besides the bigger 'Fanuc velocity', the 'stabilization time' is smaller than experiment 1) and the MSE value is bigger, as expected. It should be noted that the MSE values for the component *tilt* are significantly smaller than the *pan* for the experiment 2) and 3); this is given to the fact that those movements were made in the same direction as the *pan* component of the PTU.

Table 5.2 presents the data obtained from the experiments with the position tracking control.

Table 5.2: Results of the experiments for the position tracking control.

Experiment	5)	6)	7)	8)
Fanuc's movement profile	Movement A		Movement B	Movement C
Fanuc velocity (m/s)	$0,46$	$1,05$	$0,46$	–
Test time (s)	$8,6$	$6,7$	$8,6$	128
K_p	$0,3$	$0,3$	$0,3$	$0,3$
MSE Pan	26 600	84 569	62 101	5 009
MSE Tilt	597	597	1 379	4 121
Stabilization time (s)	$2,5$	$2,1$	$6,3$	$1,1$

Table 5.2 shows that besides the MSE values are higher then the velocity tracking control (see Table 5.1) the 'stabilization time' is lower for the experiences 5) and 6).

Table 5.3 presents the frequency of the published data during the experiments.

Table 5.3: Frequency results of the data acquisition on the control by velocity and position experiments.

Experiment	1)	2)	3)	4)	5)	6)	7)	8)
Find target node	$14,3\,Hz$	$15,2\,Hz$	$14,1\,Hz$	$15,4\,Hz$	$13,4\,Hz$	$11,7\,Hz$	$14,0\,Hz$	$15,1\,Hz$
Tracking node	$14,9\,Hz$	$15,2\,Hz$	$14,7\,Hz$	$16,4\,Hz$	$13,8\,Hz$	$12,9\,Hz$	$14,9\,Hz$	$15,5\,Hz$
Servos position	$15,3\,Hz$	$15,1\,Hz$	$14,7\,Hz$	$16,5\,Hz$	$13,6\,Hz$	$12,8\,Hz$	$14,7\,Hz$	$15,5\,Hz$
Servos velocity	$15,3\,Hz$	$15,1\,Hz$	$14,7\,Hz$	$16,5\,Hz$	–	–	–	–
Fanuc robot	$38,1\,Hz$	$37,8\,Hz$	$37,9\,Hz$	$37,0\,Hz$	$37,9\,Hz$	$38,3\,Hz$	$38,3\,Hz$	$37,1\,Hz$

Previous Table 5.3 demonstrates that there was no significant variations between the experiences, regardless of its parameters and tasks they were performing. With respect to the computer vision nodes ('tracking' and 'find target'), the difference between them is not significant and they are much inferior with regard to the tests described on Subsection 4.1.3; nevertheless, the performance of the node 'find target' is constantly inferior to the 'tracking' one.

Tables 5.4 and 5.5 are a resume of the data indicated above.

Table 5.4: Data comparison of the MSE values for experiments 1), 2), 3), 5), 6) and 7).

	Velocity Control	Position Control
Pan	103 657	173 271
Tilt	114	2 573
Sum	103 771	175 844
Percentage	37%	63%

Concerning the MSE data, the Tables demonstrate that the velocity control had a better performance; the position control is 70% higher than the velocity control in for the data presented on Table above 5.4 and 44% higher on Table 5.5.

Table 5.5: Data comparison of the MSE values for the experiments 4) and 8).

	Velocity Control	Position Control
Pan	3 626	5 009
Tilt	2 722	4 121
Sum	6 348	9 123
Percentage	41 %	59 %

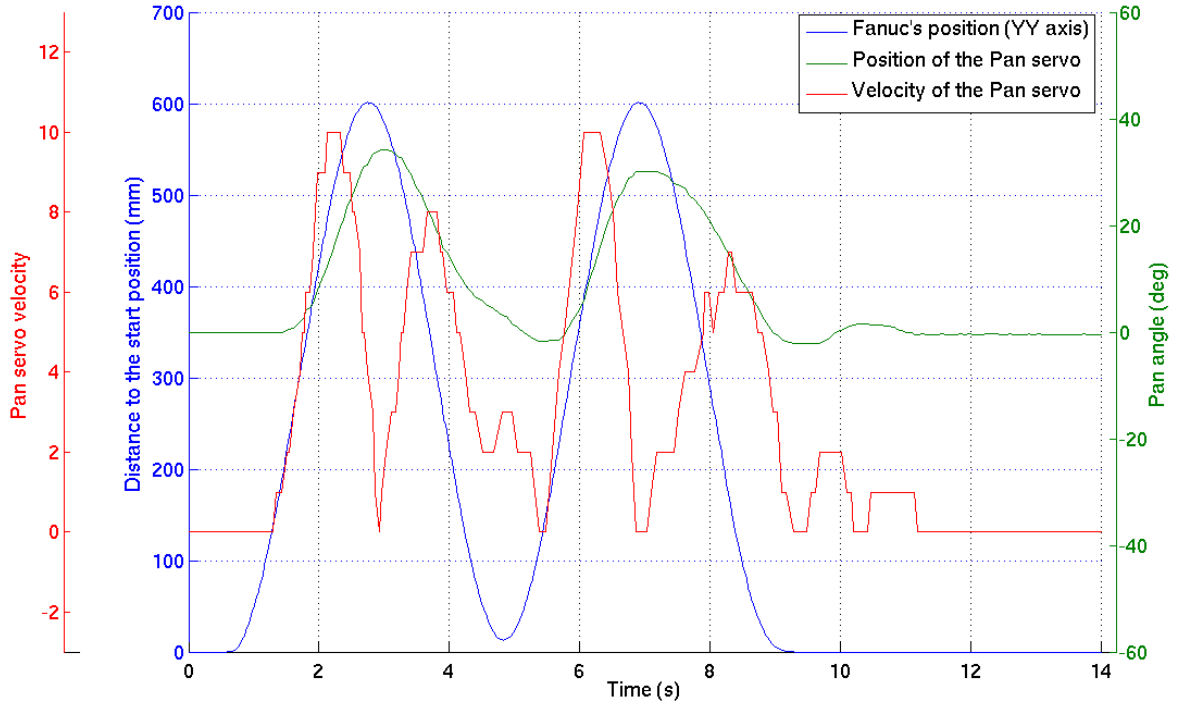
With respect to 'stabilization time', besides the difference is not pronounced, Table 5.6 shows that the time is superior in the velocity control.

Table 5.6: Comparison of the sum of the 'stabilization time' data at the end of each experiment.

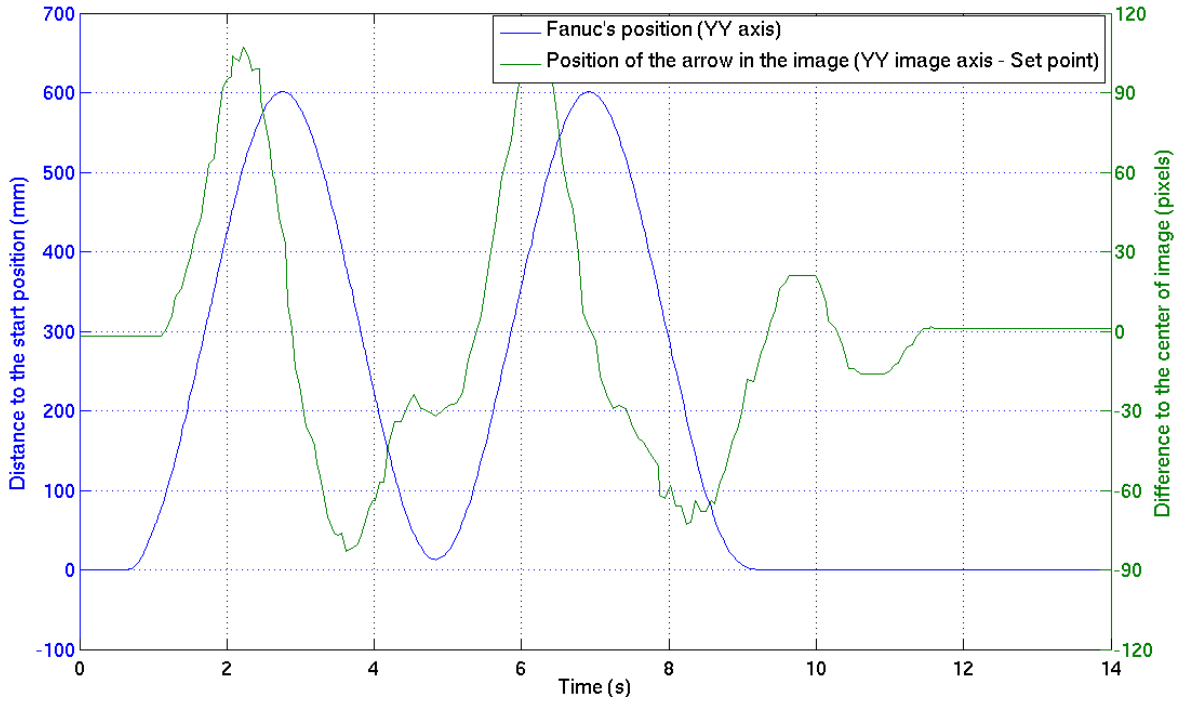
	Velocity Control	Position Control
Time sum	$15,53$	$12,02$
Percentage	$56\,\%$	$44\,\%$

The following Figures present some selected graphics of particular cases of the experiments presented above. It is noteworthy that all measurements presented bellow refer to the beginning of the experiment (e.g. the measurement of the angle of the servomotors refers to the start position of the experiment).

Figure 5.3 and Figure 5.4 are a graphical exemplification of a 'normal' experiment. This particular case refers to the experiment 3) with a velocity tracking control. The velocity data of the servomotors illustrated below are in magnitude.

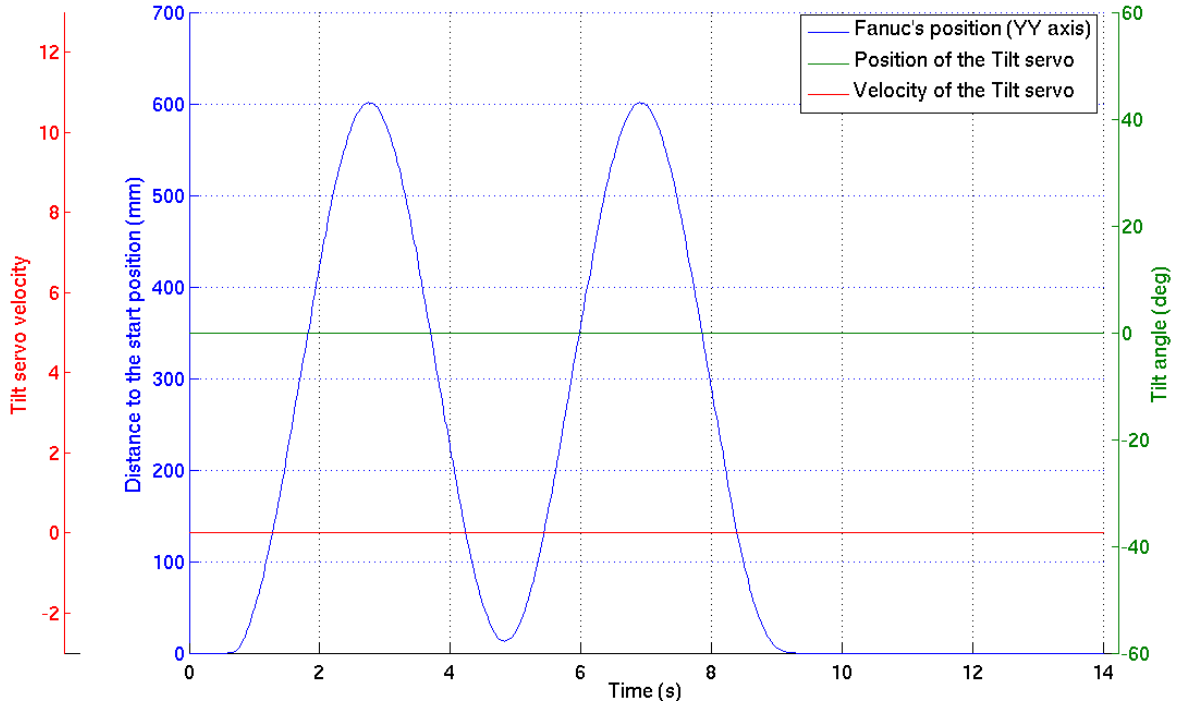


a)

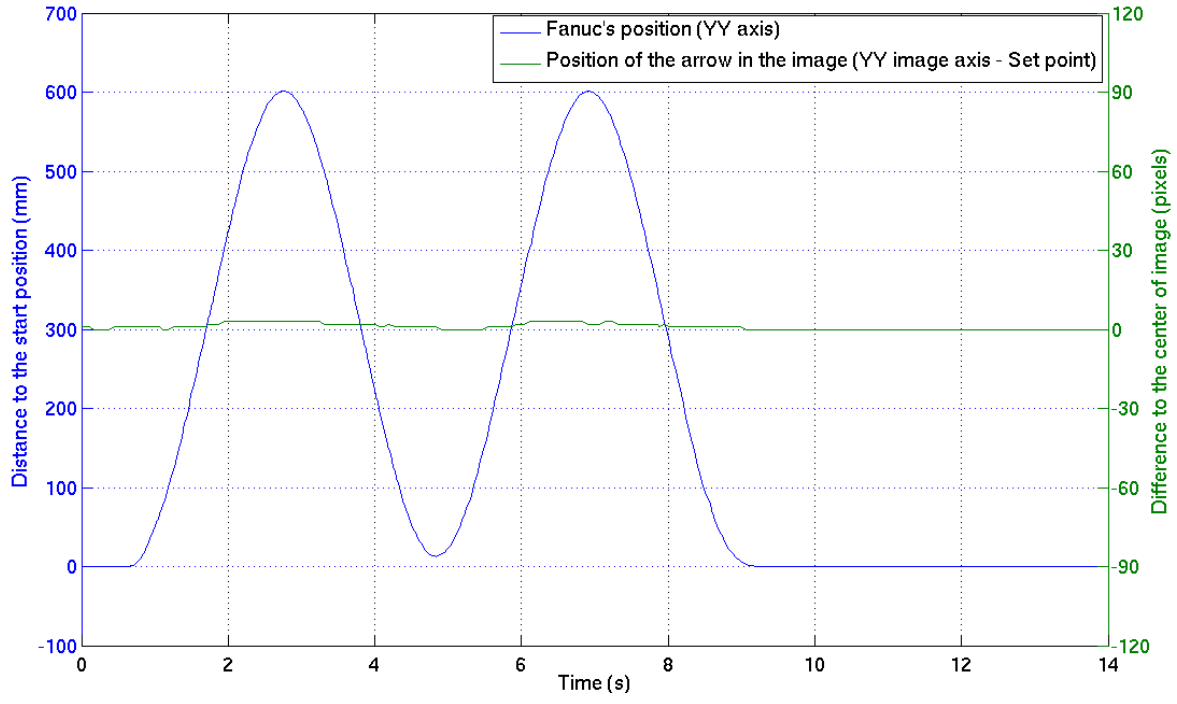


b)

Figure 5.3: Graphical output of the *pan* data acquired during the experiment 3), which is a velocity control tracking with the movement B. The graphic a) relates to the data of the PTU's servos position and velocity; b) illustrates the distance of the target in relation with the center of the image over time. Both illustrations present the Fanuc robot movement profile on the *yy* axis as a comparison "pattern".



a)

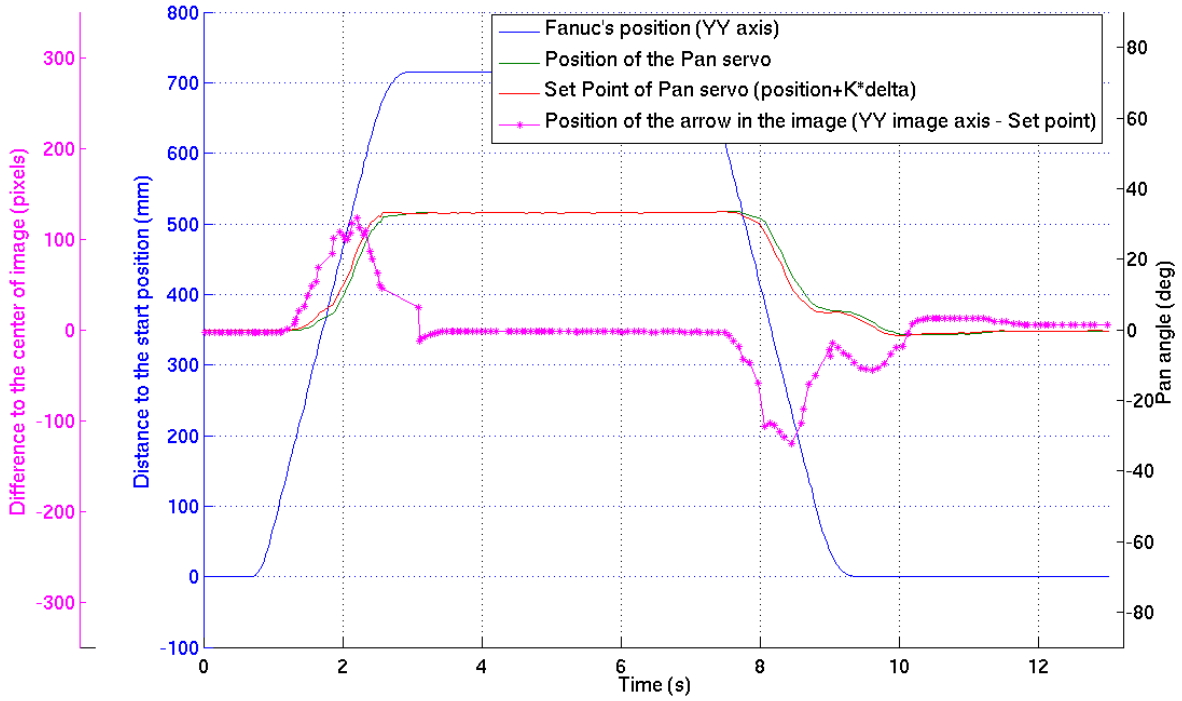


b)

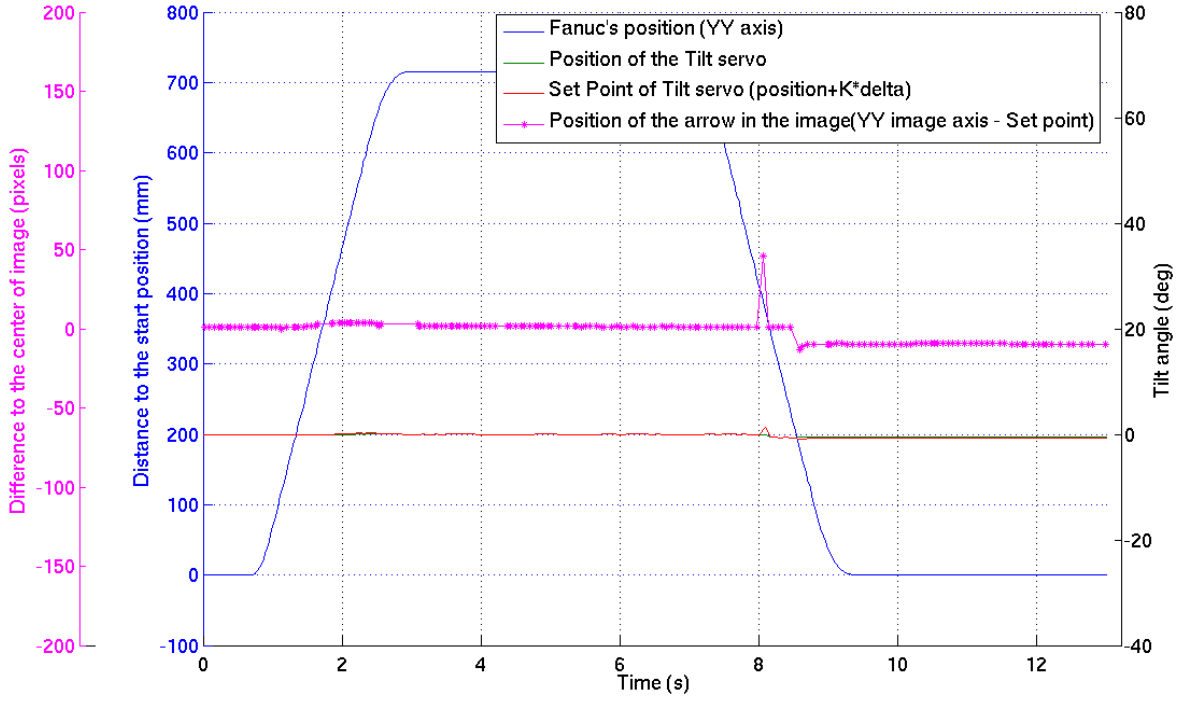
Figure 5.4: Graphical output of the *tilt* data acquired during the experiment 5), which is a position control tracking with the movement B. The graphic a) relates to the data of the PTU's servos position and velocity; b) illustrates the distance of the target in relation with the center of the image over time. Both illustrations present the Fanuc robot movement profile on the *yy* axis as a comparison "pattern".

Figure 5.3 clearly illustrates the 'stabilization time' and allows to analyze the performance of the PTU tracking system over time ($9s\,to\,11s$). Analyzing the data of the position of the PTU's servomotors versus the velocity, it is evident that the velocity is the derivative of the position, as expected. Beside the fact that the error in the image goes up to 100 pixels, the system successfully performed the tracking of the target. Figure 5.4 does not report any relevant data; since the movement on the *tilt* component is negligible, i.e, the curves remain roughly flat.

Figure 5.5 presents a graphical representation of experiment 5) with the position control tracking and performing the movement A. The red line refers to the current position of servomotor and the green line to the next SP position. It may be noted that the difference between those two lines is proportional to the distance of the target to the center of the image. In the graphic b) of the same Figure, there is a value that is an outlier. Despite of the fact that this was the only detected case on the experiments performed, it should be considered to apply an filter such as a moving average filter.



a)



b)

Figure 5.5: Graphical representation of the data acquired during the experiment 5); a) Relates to the *pan* data and b) stands for the *tilt* movement of the PTU. In both graphical representations are expressed the position of the servomotors, the SP and the error measured by the computer vision algorithms over time. Both illustrations present the Fanuc robot movement profile on the yy axis as a comparison "pattern".

With regard to the profile of the published ROS messages frequency, Figure 5.6 shows the distribution of time periods between each sent message; on this particular case it concerns to the servos position data messages. The data is from the experiment 5) (the same as previous Figure 5.5).

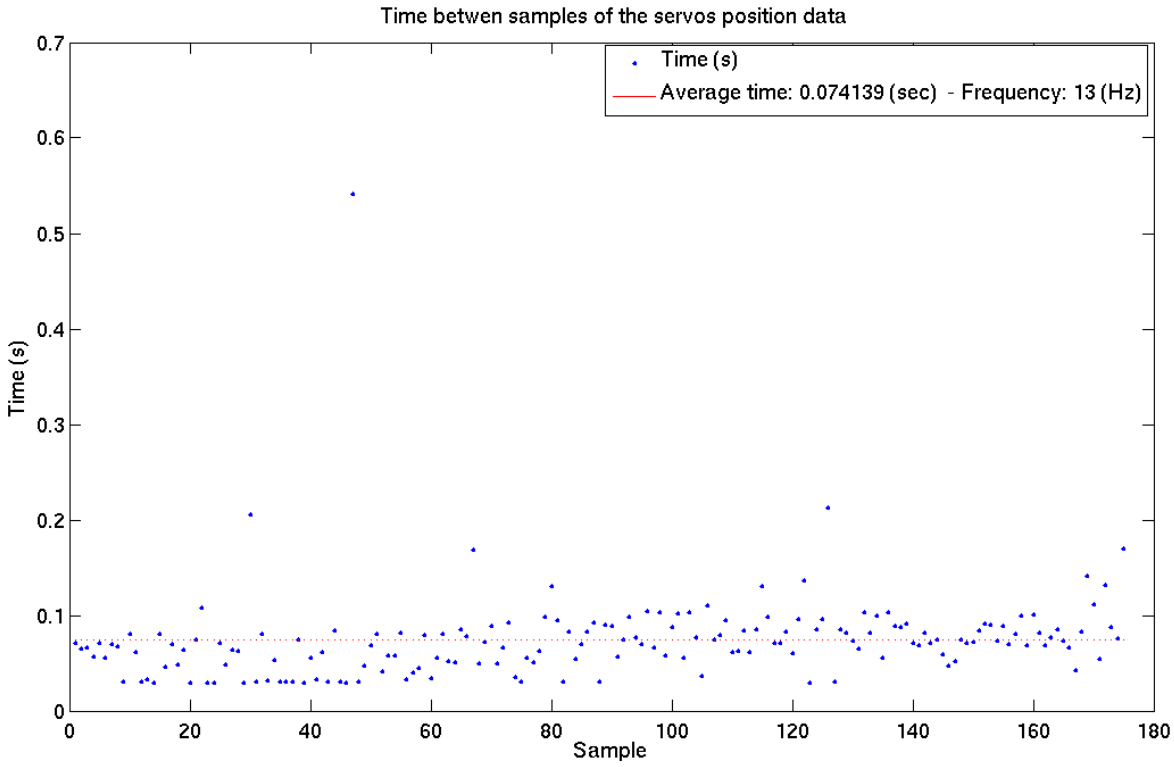
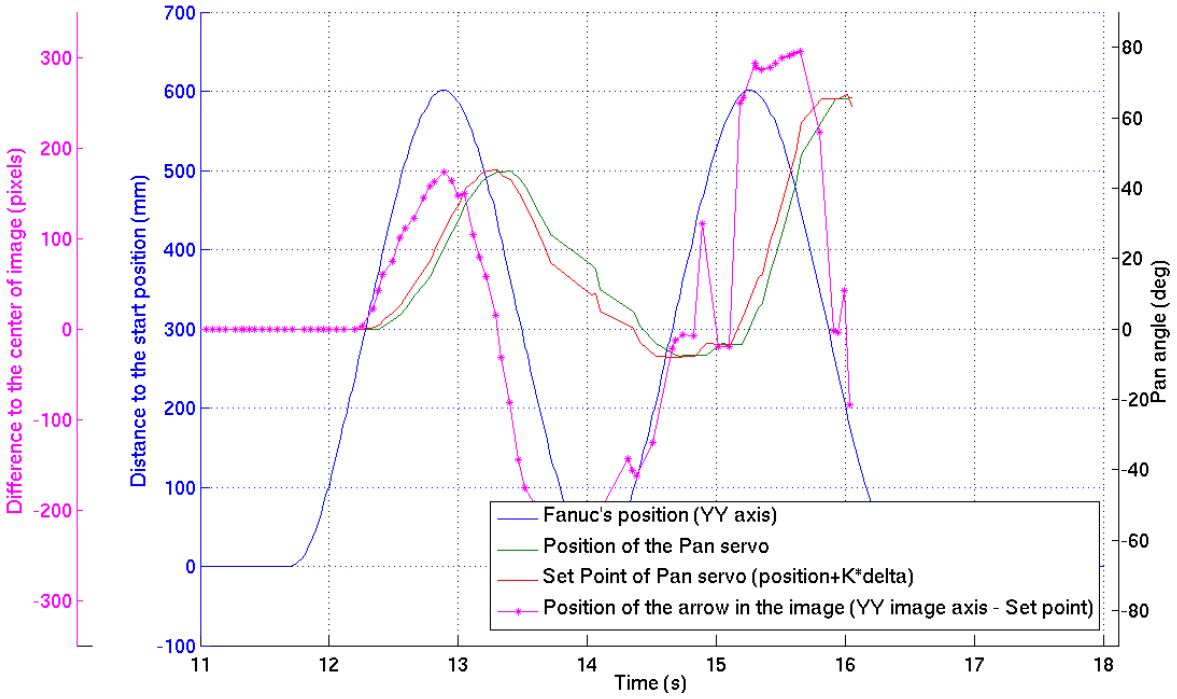


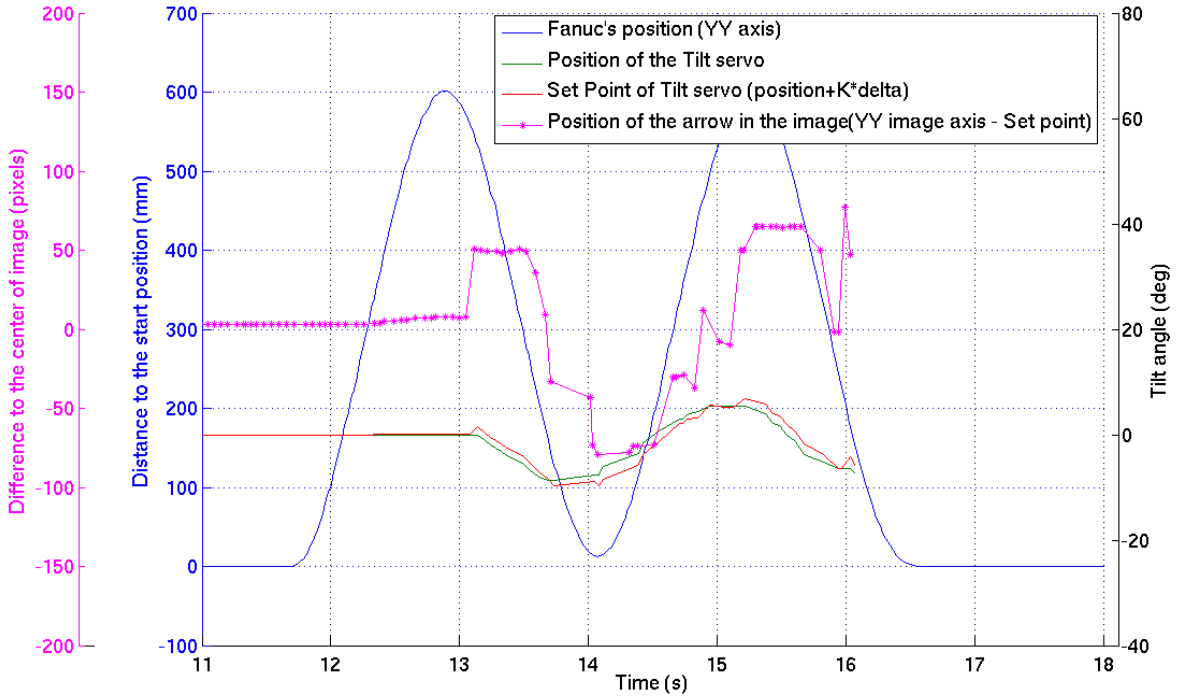
Figure 5.6: Graphical representation of the frequency of the published data by the servos controller node in the experience 5).

The results show that the time between each published ROS message data is roughly constant; nevertheless, it is noticed an exception in the sample 47 that corresponds to the time period from $19,6s$ to $20,1s$ in the previous Figure 5.5, it should be noted that, given the fact that the servomotors react to the computer vision input and they did not receive any instruction during that period.

The following Figure 5.7 presents an example of an experiment in which the target was lost by the PTU's tracking system.



a)



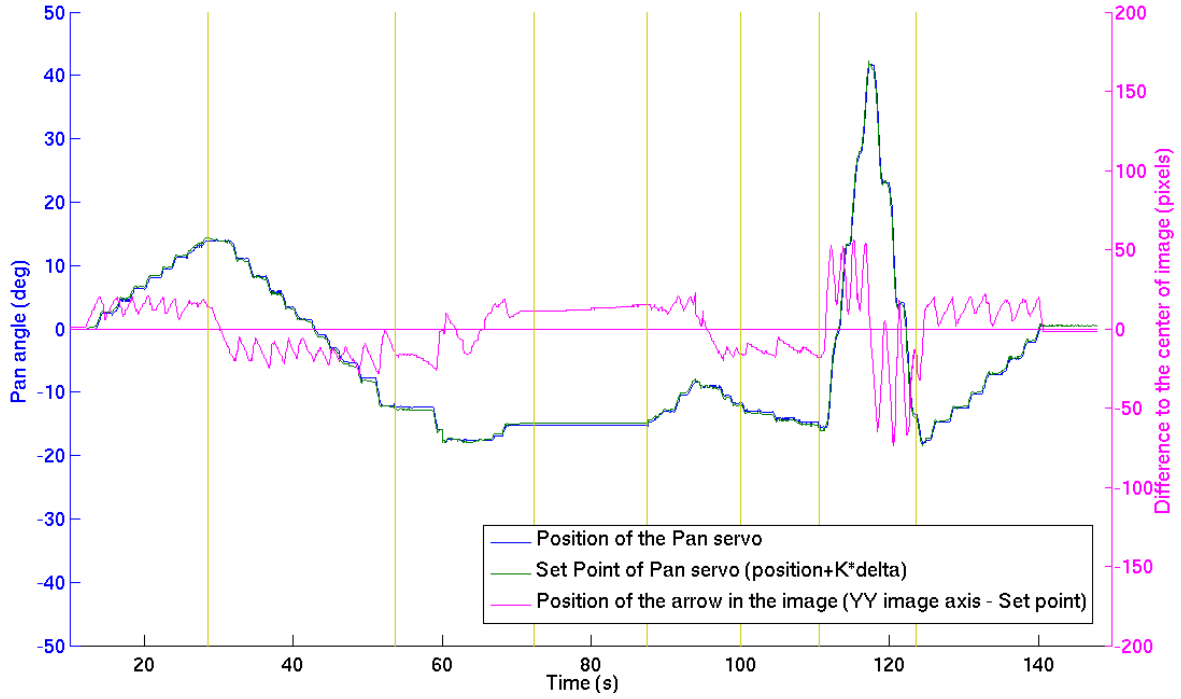
b)

Figure 5.7: Example of a failed experiment (the target was lost by the PTU) with a position tracking control and the Fanuc robot movement profile B. Graphic a) suits for the *pan* component of the PTU and b) for the *tilt*.

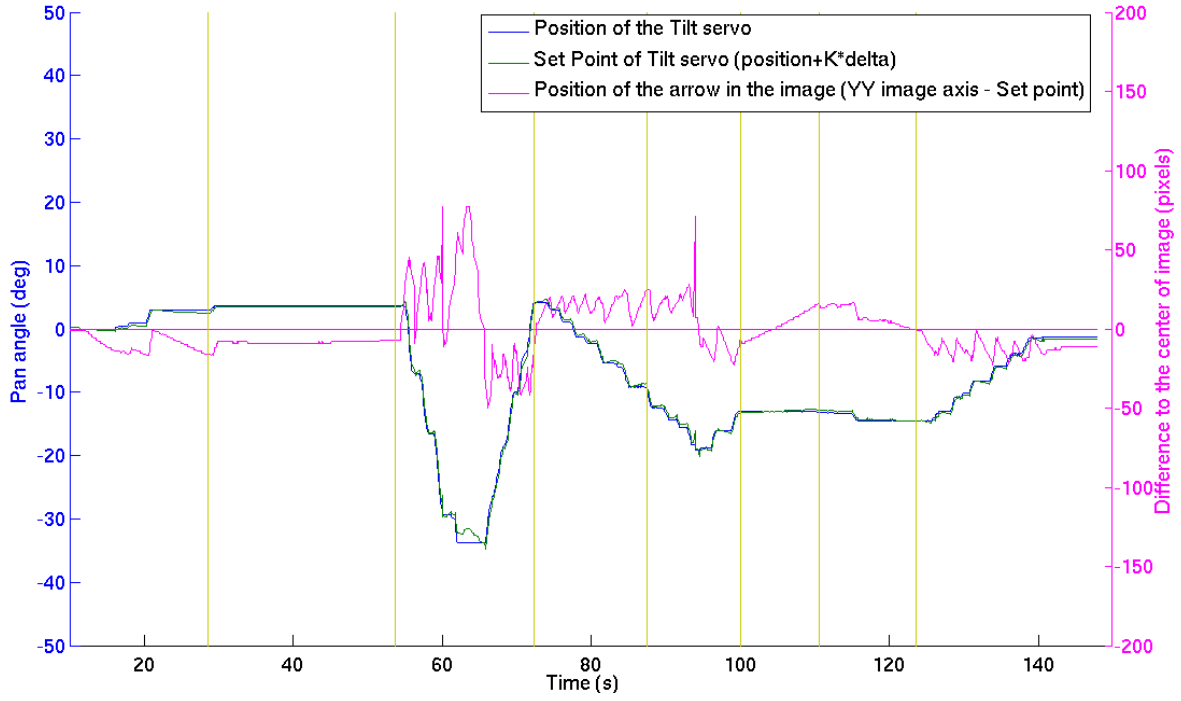
Figure 5.7 illustrates an experiment in which the position control tracking was in

operation and the movement B of the Fanuc robot. The constant K_p performing the tracking controller was $0,3$ and the maximum velocity of the Fanuc robot was roughly $1,05$.

The two following Figures have to do with the experiments 4) and 8).

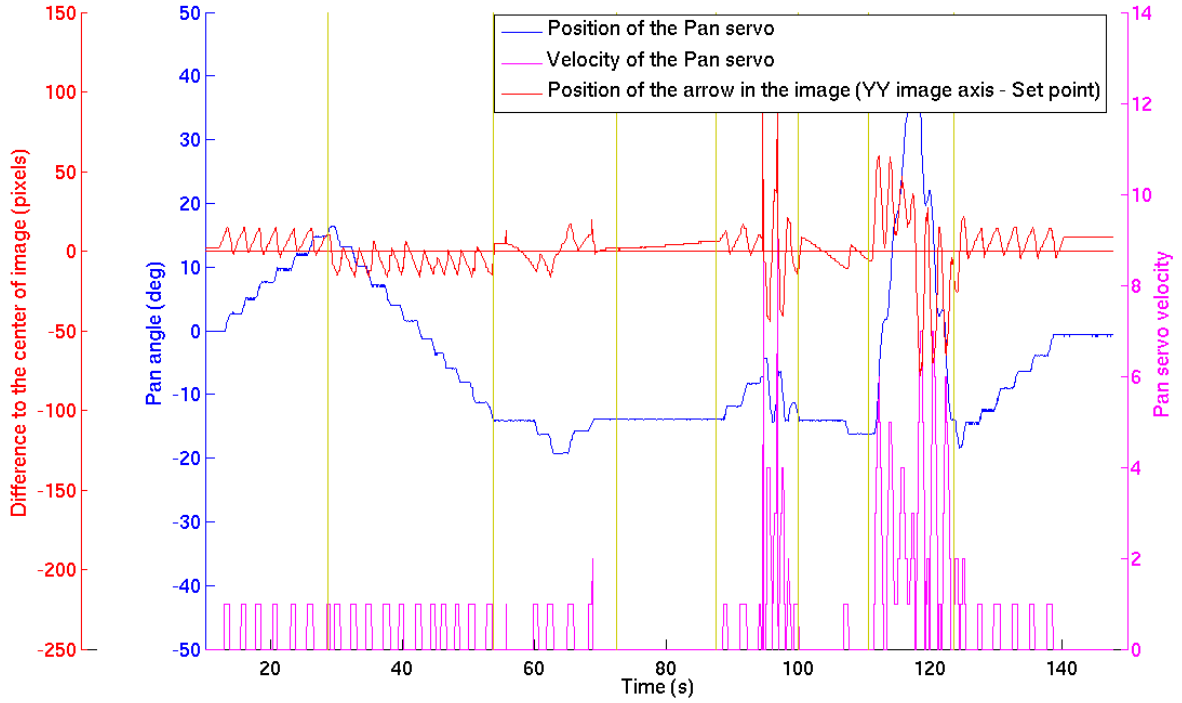


a)

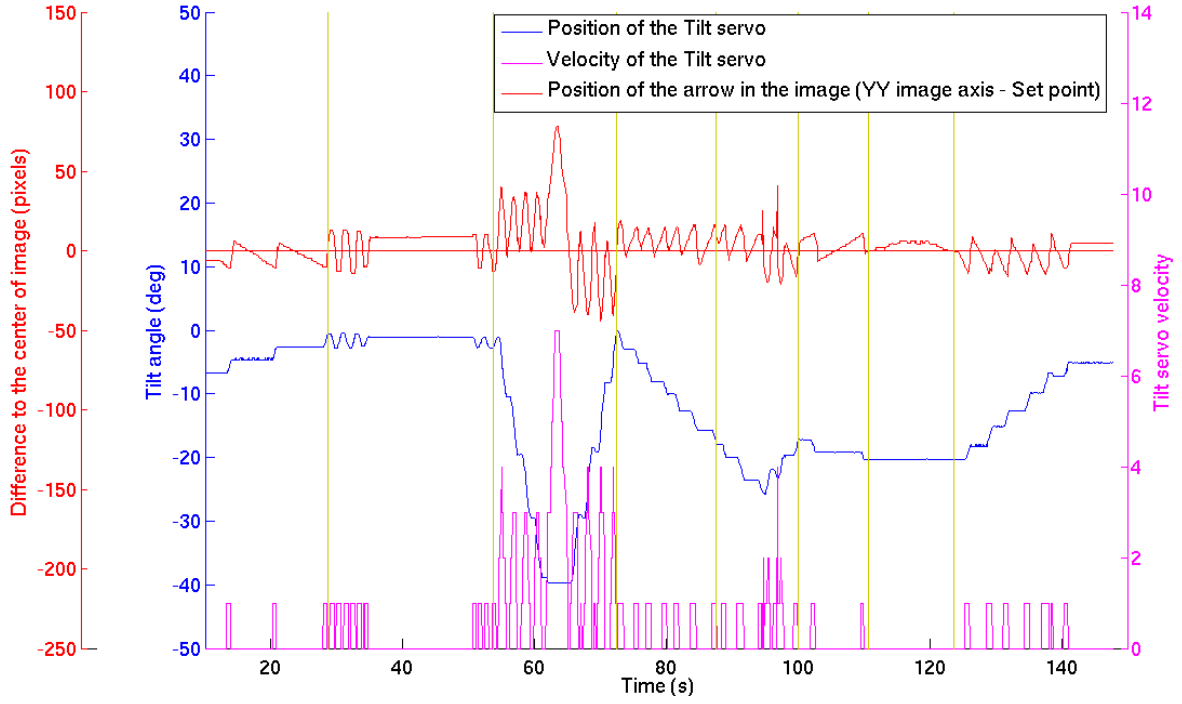


b)

Figure 5.8: Illustration of the performance of the position tracking control for experiment 8). The graphic a) suits for the *pan* component of the PTU and b) for the *tilt*.



a)



b)

Figure 5.9: Illustration of the performance of the velocity tracking control for experiment 4). The graphic a) suits for the *pan* component of the PTU and b) for the *tilt*.

Figures 5.8 and 5.9 demonstrates the expected distinctions between each segment

of the movement of the Fanuc robot. Depending on the movement at any given time (see Figure 5.2), there is one and/or another joint of the PTU that is reacting. In order to help the reader's visualization, besides the vertical yellow lines, it was also drawn an horizontal in the course of the value zero.

Besides it is a smooth difference, the best tracking controller designed was the **velocity** controller with the parameter $K_p = 0,1$.

The following section reports how a MATLAB script was designed and developed to analyze the recorded data from the experiments.

5.2 PTU Simulator

The previous described MATLAB simulator on Section 4.4 was adapted to receive the data from the Fanuc's end-effector position and the position of the servomotors obtained on the experiments described in the previous Section 5.1. With this modification, based on the Fanuc's position, it is possible to compute the SP that the control system ideally would output to command the servomotors taking advantage of the kinematic equations and the real one, measured by the servomotors encoders.

The following geometric transformation is applied to describe the relation between the global reference and the PTU's camera focal point ${}^{W}T_{FP}$, equation 5.2.

$$ {}^{W}T_{FP} = {}^{W}T_{R} \times {}^{R}T_{F} \times {}^{F}T_{E} \times {}^{E}T_{FP} \tag{5.2} $$

Since the global reference was established to be coincident with the FANUC's reference, the referential transformation ${}^{W}T_{R}$ is an identity matrix and the transformation ${}^{R}T_{F}$ is calculated by the FANUC robot. The transformation ${}^{F}T_{E}$ suits for the correlation between the FANUC's end-effector and the PTU's referential. Finally, ${}^{E}T_{FP}$ is the transformation matrix of the PTU's focal point in relation with the PTU's referential.

The simulator recognizes the target's position in relation with the global reference ${}^{W}T_{A}$ by applying the following Equation 5.3, which is required to integrate all the information to make the simulation.

$$ {}^{W}T_{A} = {}^{W}T_{R} \times {}^{R}T_{F} \times {}^{F}T_{A} \tag{5.3} $$

The ${}^{F}T_{A}$ was obtained by placing the target at one meter from a known FANUC's position on the XX axis direction.

Figure 5.10 is the graphical output of a simulated experience. In this case, it is a control by velocity experiment, with the linear movement B and a $0,559\,(m/s)$ of maximum velocity, experiment 3).

The ideal SP values based on the Fanuc's data in which the controller would calculate on the ideal control configuration is represented in a red line. The blue line represents the real path made by the servomotors and measured by themselves.

Despite the *tilt* data of the experiment exposed in Figure 5.10 has an apparent offset, it can be explained through the movement calculated by the velocity controller. Thus, depending on the proportional value K, it may not have such resolution to accomplish the illustrated difference, the *dead zone*.

The simulator also highlights the errors of the implemented control system in Figure 5.11 based on the simulated movement presented on the previous Figure 5.10 (in degrees).

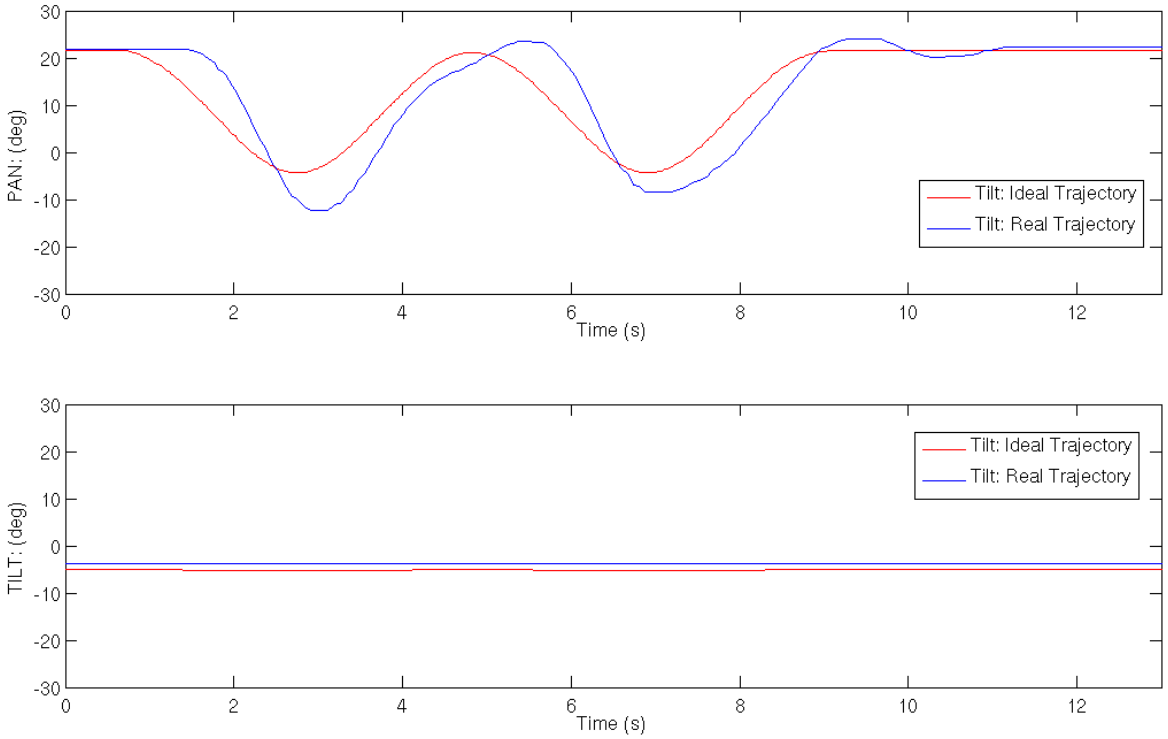


Figure 5.10: MATLAB Simulation of the control by velocity experiment with a linear movement B and 0.559 m/s of maximum Fanuc velocity.

Regarding Figure 5.11, it represents the difference between the ideal SP calculated by the MATLAB simulator which includes the kinematics equations of the PTU and the real position of the servomotors for both *pan* and *tilt* components.

By analyzing the Figure 5.11, it quickly becomes evident that the error on the *pan* movement is bigger than the *tilt* movement component. This can be explained by the fact the movement is made along with the YY axis regarding the global reference, roughly the same plane as the target and considering the $Y - X$ plane with a fixed Z. Another factor is the error introduced by the simplification described on Section 4.2 on the implemented controller: the bigger the error on the image plane, the bigger the error introduced by the controller. Still in the same movement experiment analyzed by the MATLAB simulator reported in the present section, Figure 5.12 presents the output graphics of the error of the target centroid which is projected on the image (in meters).

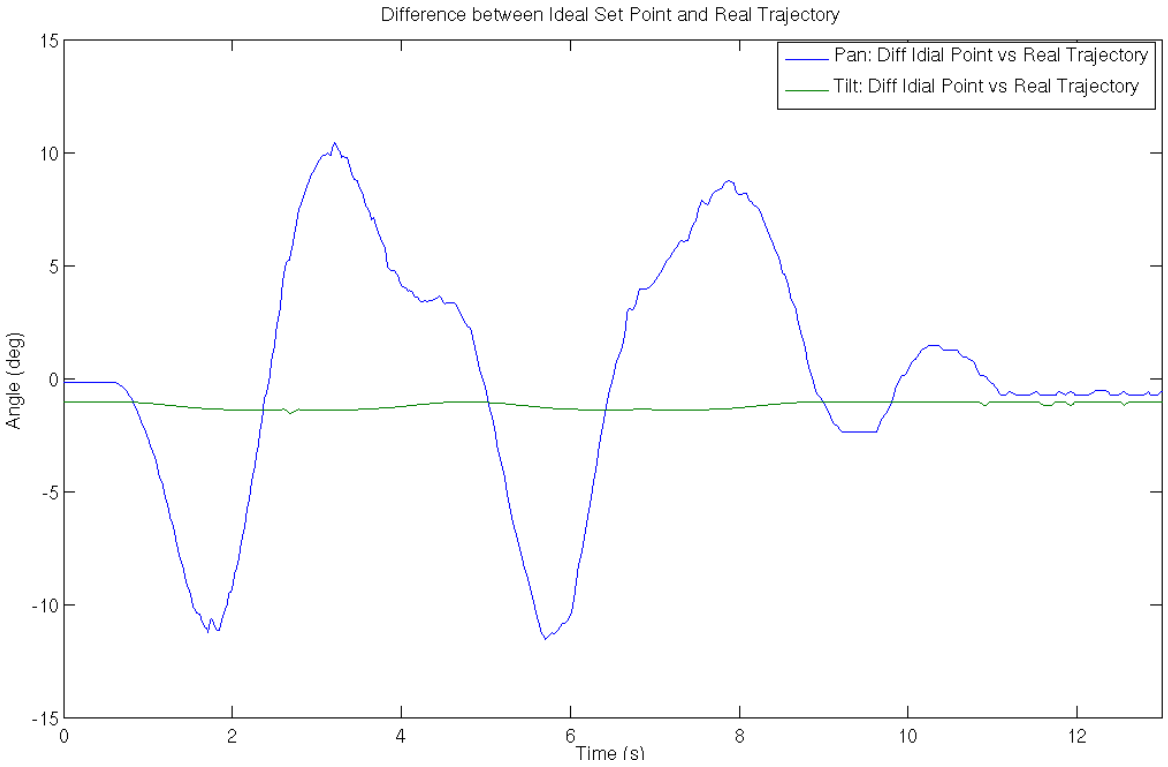


Figure 5.11: Introduced error of the PTU control system on movement.

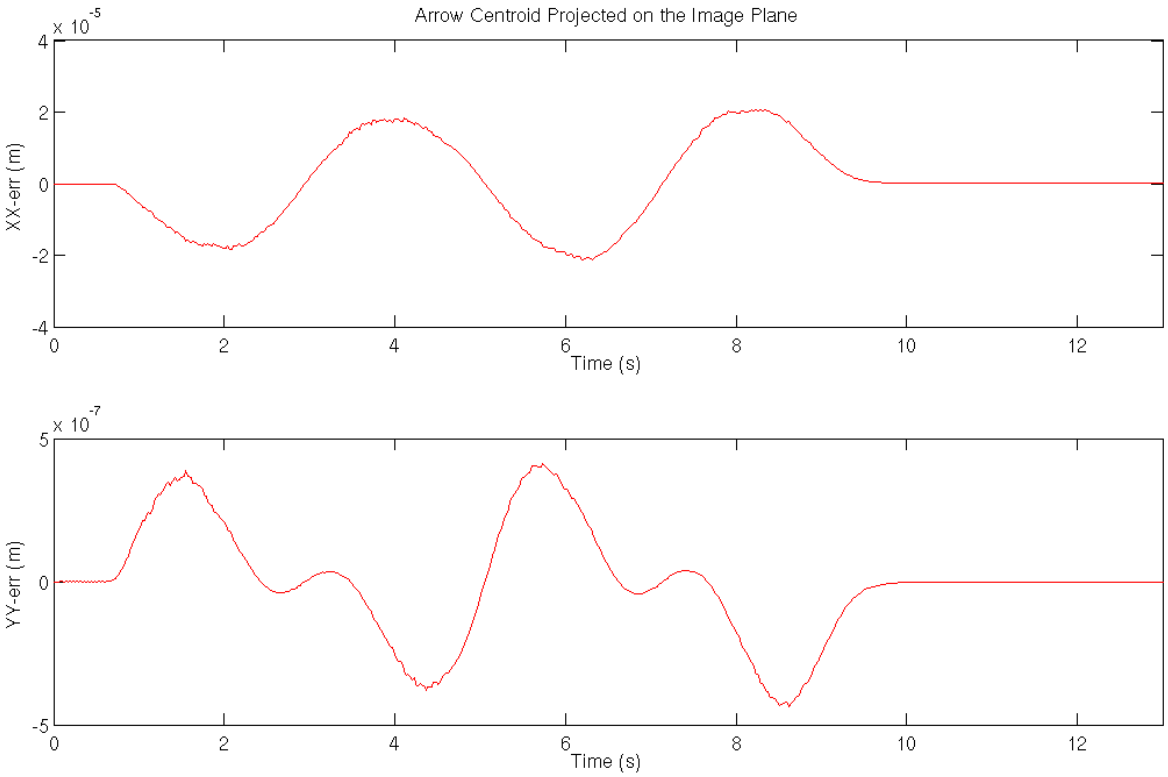


Figure 5.12: Error of the target's centroid projected on image plane.

The Table 5.7 gathers the MSE values of the conducted experiment (control by velocity with the linear movement B and a $0,559\,(meters/second)$ of maximum velocity), experiment 3).

Table 5.7: MSE values relation of the introduced errors by the control system.

	Ideal SP vs Real Trajectory		Error on image plane
Pan	$69,6$	xx	$6,9 \times 10^{-9}$
Tilt	$1,6$	yy	$1,9 \times 10^{-12}$

On the left of the above Table 5.7 it is presented the MSE values for the difference between the ideal SP and the the real position of the servos overtime. The table on the right suits for the MSE of the data presented on Figure 5.12 for the error on the image plane.

5.3 Ego-motion Estimation Analysis

This section presents the obtained results for the ego-motion estimation applied on the experiments. The results presented in this section were computed and analyzed in the previously described MATLAB simulator. It was implemented the ego-motion equations from the Section 4.3 in this simulator and integrated with the data from the experiments.

Figure 5.13 illustrates a comparison between a ego-motion estimation for ideal and real joint angles (movement A). As expected, the deviation to the ideal is proportional to the oscillation presented in the tracking of the target. Moreover, the error between the ideal and the real ego-motion estimation is increased by the fact, the distance to the target is obtained by the simulator (unsynchronized with the data from the servomotors) . This fact may be noted in the second 2 (red line). Nevertheless, the obtained results are satisfactory, suggesting that the implementation of a ego-motion estimation process is promising.

Figure 5.14 presents the velocity profile for the ego-motion estimation such as for ideal values as for the real estimation with the data from the servomotors of the PTU. It is noteworthy that, as expected the real velocity of the Fanuc and the ego-motion estimation is virtually the same.

Figure 5.15 illustrates the ego- motion estimation influence of the error introduced in the measurement of the distance of the target to the PTU. Finally, Figure 5.16 presents a comparison between the ego-motion estimation with real angles from the servomotors and the simulated ones for the zz-axis component. In this case, it is proportional to the tilt component of the PTU.

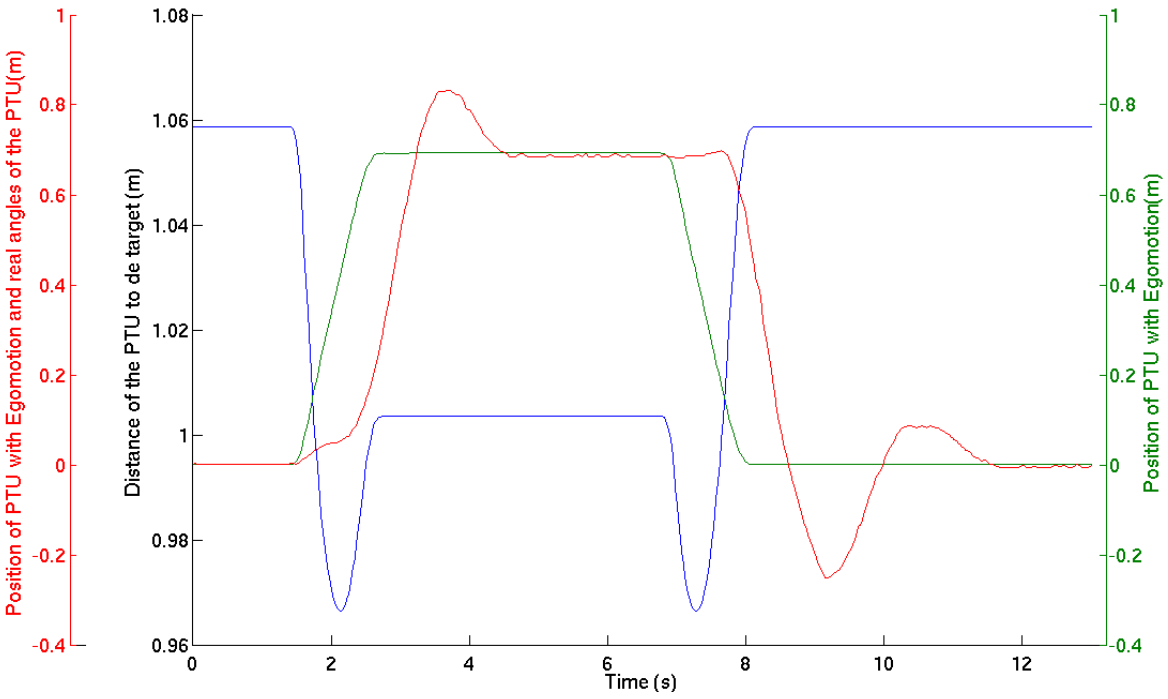


Figure 5.13: Distance of the PTU to the target in the xx axis (blue line), position of the PTU with Fanuc data (green line) and ego-motion estimation of the position of the PTU with the real angles obtained from the PTU's servomotors (red line).

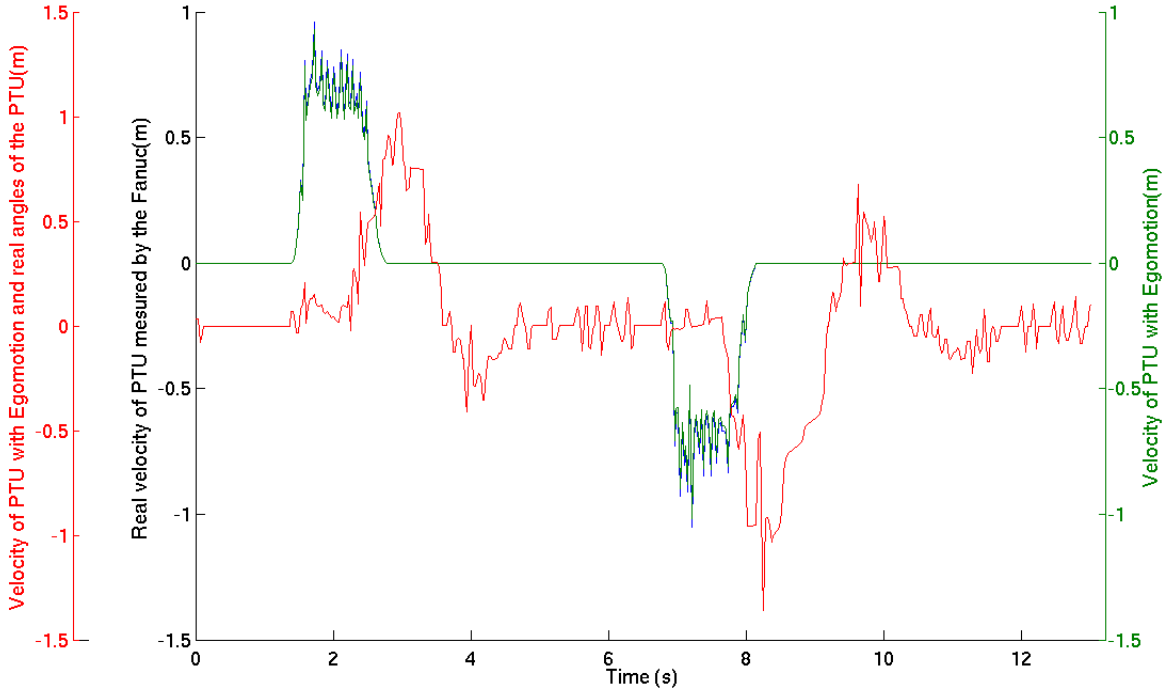


Figure 5.14: Velocity profile of the PTU and its ego-motion estimation (blue and green line respectively) and the ego-motion estimation velocity with real angles obtained from the PTU's servomotors (red line).

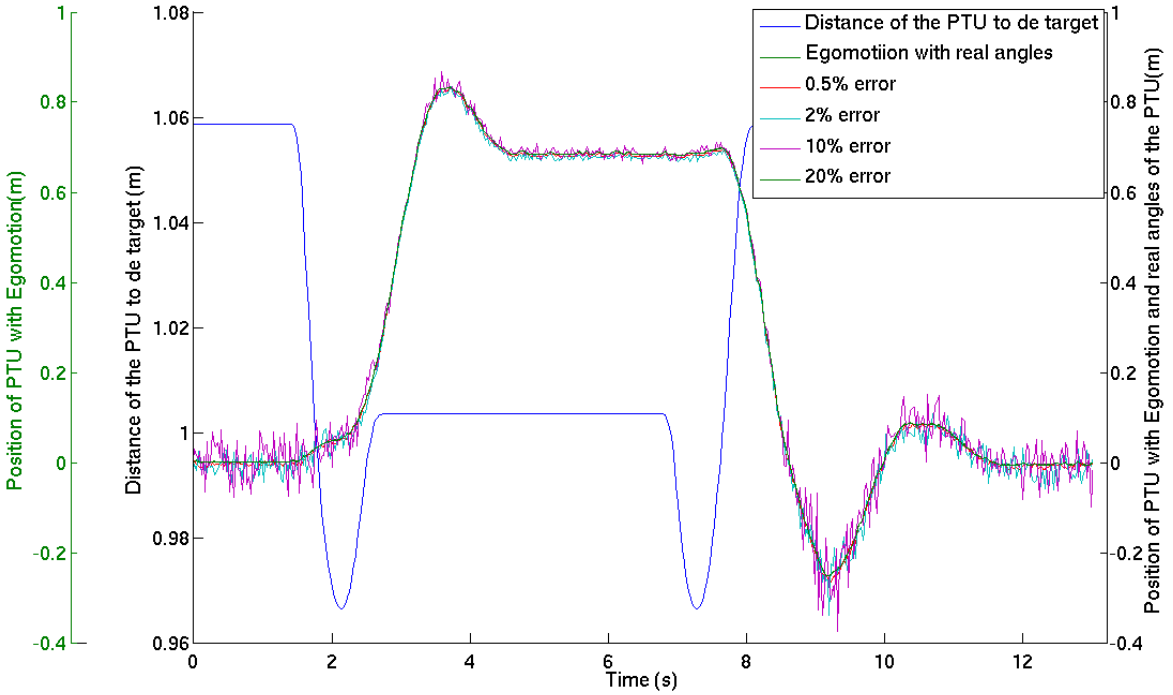


Figure 5.15: Distance of the PTU to the target in the xx-axis (blue line), ego-motion estimation of the position of the PTU with the real angles obtained from the PTU's servomotors (red line) and comparison with error introduced in the distance.

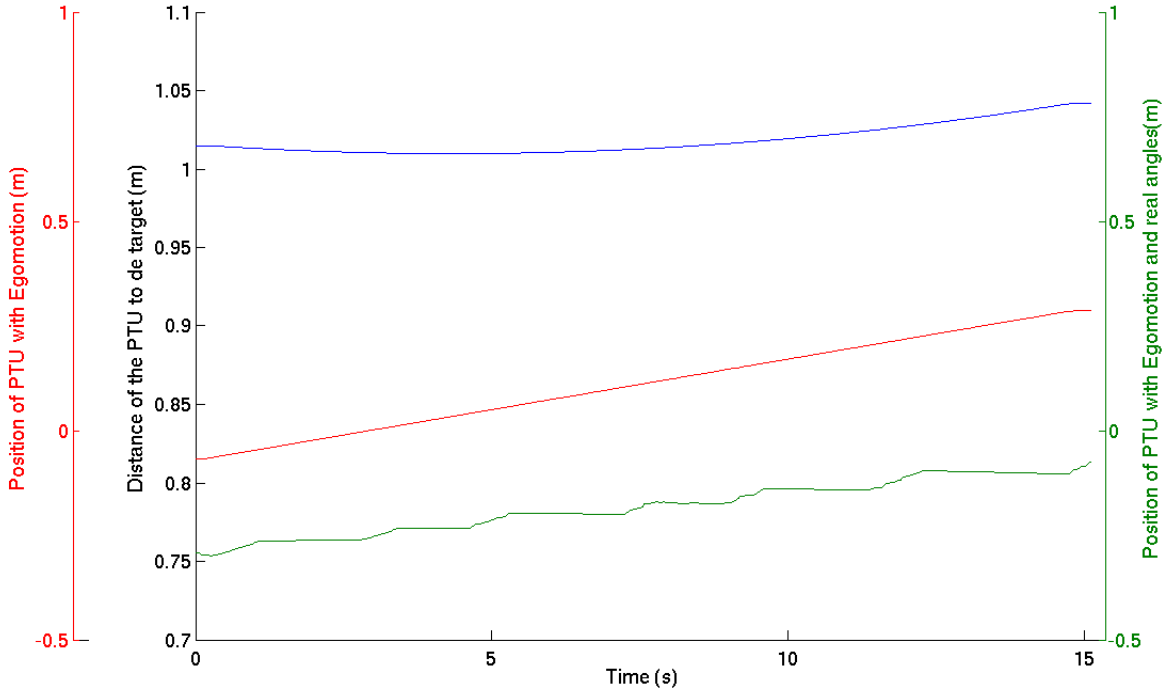


Figure 5.16: Distance of the PTU to the target in the zz-axis (blue line), position of the PTU with Fanuc data (red line) and ego-motion estimation of the position of the PTU with the real angles obtained from the PTU's servomotors (green line).

5.4 Discussion

The previous referred ideal control system covers the kinematics of the PTU and the target's distance to compute the SP on each given moment. In order to study all the parameters involved on the equation, Figure 5.17 is used to illustrate a movement of the PTU on a given moment exactly when the camera captures the target on the image[2]. The Figure also illustrates the position where the camera captures the image within the target on the center of it ($C2$).

[2]In this particular case, the target is on the top right of the captured image.

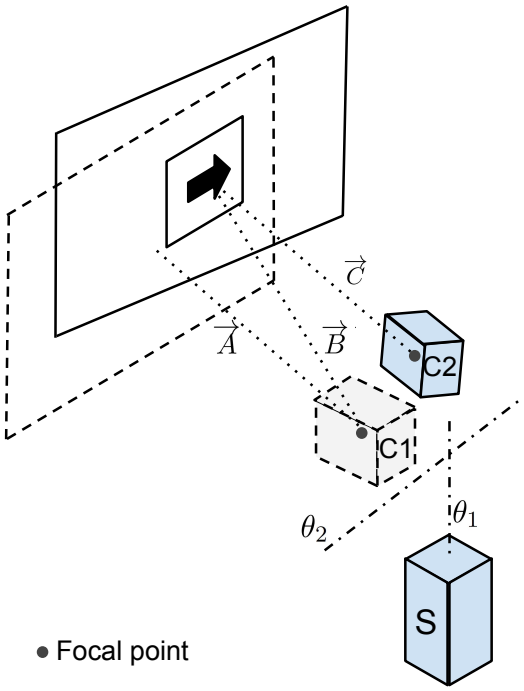


Figure 5.17: Illustration of a PTU's movement from an given point gathering the target's direction, towards a position where the camera is faced with the target in the center of the image.

In the first instance, in the position $C1$, the camera records the image within the target on the top right of the image frame. The vector $\overrightarrow{A}$ represents the ZZ axis of the camera, starting in the $C1$ camera's focal point towards the center of the FOV, and the vector $\overrightarrow{B}$ starts in the $C1$ camera's focal point towards the target's center.

At this point, the controller must compute the data in order to discover the *pan* and *tilt* angles θ_1 and θ_2, respectively, intended to reach the position $C2$ where the vector $\overrightarrow{A}$ assumes the same profile as vector $\overrightarrow{B}$ reaching the vector $\overrightarrow{C}$. In other words, the target is on the middle of the image frame. With the previous given information it is not possible to move the PTU's camera to such direction. When $\overrightarrow{B}$ is in the same location as the position $C1$, the camera is rotated around the focal point in a such way that the vector $\overrightarrow{A}$ is coincident with vector $\overrightarrow{B}$. Therefore, the target would be at the middle of the image and the camera on position $C1$.

The PTU only has two DOF; for a specific position of its end-effector[3], it has one and only one direction. Consequently, the PTU's controller must find a second position ($C2$). To find such position, i.e, the vector $\overrightarrow{C}$, it is necessary to find or get the position of the target in the space.

[3]Focal Point.

At this point, the information that is known about the target at the concerned iteration ($C1$) is the direction $\overrightarrow{B}$ for the given PTU angles θ_1 and θ_2; along with the vector $\overrightarrow{B}$, it is missing the target's position (in this vector).

The solution that the author proposes to get the position of the target on the vector $\overrightarrow{B}$, taking into consideration that target does not change its shape and size over time and the target's blob area changes with its position along with the vector $\overrightarrow{B}$ on the image, consists of the extraction of the target blob's area. Moreover, the blob contour data is easily extracted and available on the implemented image processing algorithm[4].

Up to this point, the PTU's reference regarding the target one is known. Therefore, based on the PTU's kinematics and the lens focal length, it is feasible to get the PTU's θ_1 and θ_2 that will direct the camera towards the target and extrapolate the information for the PHUA's balance and navigation system. However, there is one simplification that can be made in order to simplify the previous described process, which is the assumption that the target's distance when comparing to the range of the PTU's movement, the direction of the vector $\overrightarrow{B}$ would be nearly parallel to the vector $\overrightarrow{C}$. This means that it would not be necessary to calculate the targets distance, as a result, the direction of $\overrightarrow{C}$ is the same as $\overrightarrow{B}$.

Another way to simplify the algorithm without compromising its performance is to improve the PTU's infrastructure with the intention to guarantee that both *pan* and *tilt* axis have its rotational axis along with the camera's focal point. Figure 5.18 illustrates the proposal of the author based on the gyroscope principle on a) inspired on the mechanical structure presented on b) [71].

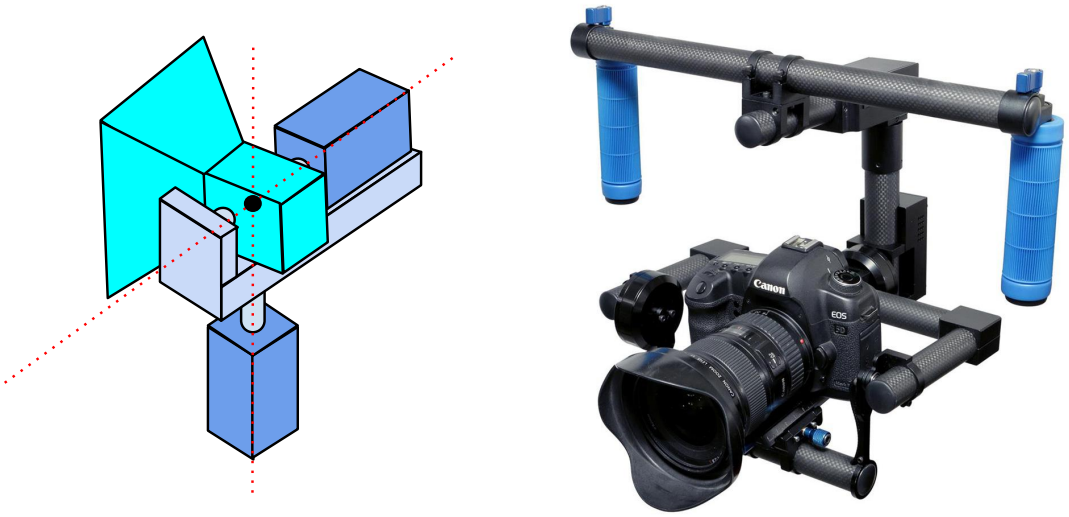
Nevertheless, this configuration has the advantage that the structure's weight is not equally distributed, unlike the one proposed on Section 3.1.

Besides the described limits that the PTU's may have to improve in the future, regarding the specific context of the PHUA's project and the application of the PTU, the movements that the PTU is expected to do would be significantly smoother then the performed tests where such situations in the context of the PHUA's balance would mean it would almost reach a free fall crossing the returning to the standing position point. Nevertheless, it has space for further improvements.

It is important to remind and record that the described experiments above and the results were made in an computer that, by the time of the present dissertation, is bellow the average of the common computer processors.

With regard to the data obtained with the PTU's visual tracking system described

[4](see Section 4.1.1)



a) b)

Figure 5.18: a) New PTU's mechanical structure proposal, b) 2-Axis Digital Gyro Stabilizer made by the company Adorama .

here that may be integrated with the humanoid balance control, the following information is obtained:

- When the humanoid knows its own position on the space as well as the position of the targets on the scene, the PTU with the visual tracking reveals the rotation of the humanoid in relation to the target;
- By the fact that the PTU is constantly staring at the target and, once again, the position of the target is known in the space, the humanoid may assess its own posture, measuring the angle of the floor plan with the *tilt* component of the PTU.

Hereby, the PTU reveals important and reliable information for the humanoid to integrate with is own sensorial system, as well as the kinematics computation during the balance and navigation process making this process more robust and redundant.

Regarding the upgraded PTU's mechanical structure, considering the large number of experiments conducted during the development of the present dissertation, it has proved to be robust and suits all the needs, considering the mechanical movements and compactness.

Chapter 6

Conclusions

The central proposition of the thesis supported by the present dissertation claims that vision can play a major role in humanoid's balance and navigation systems. A state of the art review was made in both humans and human-like robots and on both cases, it was found clear evidences that support the proposed thesis. It was also made an empirical experiment on a laboratory equipped with a stereo vision system VICON and a force platform that compared the balance of a human subject with the eyes open and cosed. The results suggested that vision plays a major role in humans balance, as predicted.

A new mechanical infrastructure was made for PHUA's neck and it did not presented any problems during the experimental tests. This infrastructure offers the possibility to be easily adapted to use two cameras (stereo vision) or increase the range of the *tilt* movement of the neck, providing a solid and stable structure for the future. It was presented a possible improvement compromising the structure's simplicity in order to simplify its kinematics and therefore, potentially improve the computational performance.

The implementation of OpenCV, ViSP and ROS was successfully accomplished, working now together in this project. Besides the use of OpenCV in this project is not strictly needed given the existent open-source software that detect patterns, it has revealed in the academic point of view, a major improvement to the author's knowledge, which is one of the objectives of this dissertation subject. The tracking algorithm based on ViSP was successfully implemented and proved to be reliable and fast. Related with the algorithm to detect the target, it has to be improved. Nevertheless, it goes out of the scope of this research, which the objective was to track one object and give valuable visual feedback to the humanoid robot. Moreover, in future, the objective would be to make an autonomous target selection based on real robot environment, without artificial landmarks.

It was developed a MATLAB simulator to predict and calculate the best PTU's trajectory based on its position and the position of the target in relation to the world. Moreover, with regard to the ego-motion estimation, it was successfully implemented the equations that obtains the position of a robot based on the angle of the PTU and the distance to the target in the MATLAB script.

The control of the servomotors is made with a simple proportional control, a more powerful computational capabilities must be implemented, nevertheless, given the actual performance and the processing velocity, it fits the needs providing a fast tracking system.

During the experiments, it was developed an C++ package to interact with the robot Fanuc that may be further used and integrated in other future projects to control the robot using Linux and ROS.

The results showed that the computer vision algorithms have a good performance for specific needs and the velocity control algorithm for the tracking system suits the best to accomplish a good and simple tracking system infrastructure in order to obtain visual feedback for the humanoid. The ego-motion simulation and results demonstrate that it is possible and reliable to concatenate this simple visual odometry system to the navigation system of a robotic system. Therefore, it is demonstrated the validity of this novel approach.

6.1 Future Work

This work opened several investigation lines for further research. Some suggestions for additional improvements and research, considering the developed work, are described next:

- Integration of this dissertation work with the existent balance control system;
- Implement an autonomous target selection, may follow the line that the flowing references suggest [15, 48, 72];
- Modulate the PTU's structure in CATIA® software, to be accorded to the other parts of the PHUA which are drawn on this software;
- Consider to improve the PTU's tracking system with a Proportional-Integral-Derivative controller (PID);
- Implement a ROS service protocol in order to the visual tracking system request the target position only when necessary;
- Test the performance of the developed system (tracking control and ego-motion estimation) with a better computer processor and the humanoid robot PHUA itself making the movements;

- Implement a stereo vision system to obtain the distance of objects and other features.

References

[1] Jung-Yup Kim, Ill-Woo Park, and Jun-Ho Oh. Walking control algorithm of biped humanoid robot on uneven and inclined floor. *Journal of Intelligent and Robotic Systems*, 48(4):457–484, 2007. ISSN 0921-0296. doi: 10.1007/s10846-006-9107-8. URL http://dx.doi.org/10.1007/s10846-006-9107-8.

[2] Jack Arnold. Creature from the black lagoon. http://www.imdb.com/title/tt0046876/, 1954. consulted at 12-8-2014.

[3] Robotics Business Review. http://www.roboticsbusinessreview.com/companies/category/humanoid/type. consulted at 12-8-2014.

[4] Yoseph Bar-Cohen, Adi Marom, and David Hanson. *The Coming Robot Revolution: Expectations and Fears About Emerging Intelligent, Humanlike Machines*. Springer, 2009.

[5] William Bluethmann, Robert Ambrose, Myron Diftler, Scott Askew, Eric Huber, Michael Goza, Fredrik Rehnmark, Chris Lovchik, and Darby Magruder. Robonaut: A robot designed to work with humans in space. *Autonomous Robots*, 14(2-3): 179–197, March 2003. ISSN 0929-5593, 1573-7527. doi: 10.1023/A:1022231703061. URL http://link.springer.com/article/10.1023/A%3A1022231703061.

[6] David Gouaillier, Vincent Hugel, Pierre Blazevic, Chris Kilner, Jérôme Monceaux, Pascal Lafourcade, Brice Marnier, Julien Serre, and Bruno Maisonnier. Mechatronic design of nao humanoid. In *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, ICRA'09, pages 2124–2129, Piscataway, NJ, USA, 2009. IEEE Press. ISBN 978-1-4244-2788-8. URL http://dl.acm.org/citation.cfm?id=1703775.1703795.

[7] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of honda humanoid robot. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 2, pages 1321–1326 vol.2, May 1998. doi: 10.1109/ROBOT.1998.677288.

[8] Aldebaran Robotics. Nao. http://www.aldebaran-robotics.com. consulted at 12-8-2014.

[9] Honda. Nao. http://asimo.honda.com/. consulted at 12-8-2014.

[10] NASA. Robonaut. http://mag.amazing-kids.org/. consulted at 15-8-2014.

[11] S. Ushida and K. Deguchi. *Vision-based Motion Control of a Biped Robot Using 2 DOF Gaze Control Structure*. INTECH Open Access Publisher, 2007. ISBN 9783902613073. URL http://books.google.pt/books?id=oLqloAEACAAJ.

[12] Saeid Pashazadeh and Saeed Saeedvand. Modelling of walking humanoid robot with capability of floor detection and dynamic balancing using colored petri net. *International Journal in Foundations of Computer Science & Technology (IJFCST)*, 4(2):1–10, 2014. URL http://arxiv.org/pdf/1404.2892.pdf.

[13] Wikipedia. http://en.wikipedia.org/. consulted at 14-8-2014.

[14] T. Takenaka, T. Matsumoto, and T. Yoshiike. Real time motion generation and control for biped robot -1st report: Walking gait pattern generation-. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 1084–1091, Oct 2009. doi: 10.1109/IROS.2009.5354662.

[15] N. Oda, J. Yoneda, and T. Abe. Visual feedback control of zmp for biped walking robot. In *IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society*, pages 4543–4548, Nov 2011. doi: 10.1109/IECON.2011.6120058.

[16] N. Oda and J. Yoneda. Visual feedback control based on optical flow vector field for biped walking robot. In *Mechatronics (ICM), 2013 IEEE International Conference on*, pages 635–640, Feb 2013. doi: 10.1109/ICMECH.2013.6519116.

[17] Filipe Silva and Victor Santos. Visual feedback to assist humanoid balance. PhD Proposal, Institute of Electronics and Telematics Engineering of Aveiro, University of Aveiro, Portugal, January 2013.

[18] Filipe Silva and Victor Santos. Multipurpose low-cost humanoid platform and modular control software development. In Matthias Hackel, editor, *Humanoid Robots, Human-like Machines*. I-Tech Education and Publishing, June 2007. ISBN 978-3-902613-07-3. URL http://www.intechopen.com/books/howtoreference/humanoid_robots_human_like_machines/multipurpose_low-cost_humanoid_platform_and_modular_control_software_development.

[19] Gary Bradski. The opencv library. *Doctor Dobbs Journal*, 25(11):120–126, 2000.

[20] Eric Marchand, Fabien Spindler, and François Chaumette. Visp for visual servoing: a generic software platform with a wide class of robot control skills. *Robotics & Automation Magazine, IEEE*, 12(4):40–52, 2005.

[21] Ricardo Costa Godinho. Desenvolvimento do tronco e braços de uma plataforma humanoíde híbrida. Master's thesis, University of Aveiro, 2011. Master Thesis.

[22] William Eras Lage. Algoritmos de controlo do movimento para um robÃŽ humanóide. Master's thesis, University of Aveiro, 2011. Master Thesis.

[23] DARREN SAWICZ. Hobby servo fundamentals. `http://www.princeton.edu/~mae412/TEXT/NTRAK2002/292-302.pdf`.

[24] Pedro Miguel Batista Cruz. Haptic interface data acquisition system. Master's thesis, University of Aveiro, 2012. Master Thesis.

[25] Telmo Filipe Jesus Rafeiro. Rede de sensores inerciais para equilíbrio de um robÃŽ humanóide. Master's thesis, University of Aveiro, 2012. Master Thesis.

[26] M. Rodrigues. Unidade de processamento e sistema de visão para um robô humanóide. Master's thesis, University of Aveiro, 2008.

[27] Alexandra S. Pollock, Brian R. Durward, Philip J. Rowe, and John P. Paul. What is balance? *Clinical Rehabilitation*, 14(4):402–406, April 2000. ISSN 0269-2155, 1477-0873. doi: 10.1191/0269215500cr342oa. URL `http://cre.sagepub.com/content/14/4/402`.

[28] Sanders T. Valeria C. Scanlon. *Essentials of Anatomy and Physiology*. F. A. Davis Company, fifth edition, 2007.

[29] Thomas M. Jessell Eric R. Kandel, James H. Schwartz. *Principles of Neural Science*. McGraw-Hill, fourth edition, 2000.

[30] Canadian cancer society. `http://www.cancer.ca/`, August 2014. URL `http://www.cancer.ca/`.

[31] Georgeson M. A. Bruce V., Green P.R. *Visual Perception*. Psycology Press, fourth edition, 2003.

[32] Fox S.I. *Human Physiology*. McGraw-Hill, eighth edition, 2003. Daniela book.

[33] G. Westheimer and S. P. McKee. Visual acuity in the presence of retinal-image motion. *Journal of the Optical Society of America*, 65(7):847–850, July 1975. ISSN 0030-3941.

[34] Hall J. E. Guyton A.C. *Textbook of Medical Physiology*. Elsevier Saunders, eleventh edition, 2006.

[35] V. J. Wilson, R. Boyle, K. Fukushima, P. K. Rose, Y. Shinoda, Y. Sugiuchi, and Y. Uchino. The vestibulocollic reflex. *Journal of Vestibular Research: Equilibrium & Orientation*, 5(3):147–170, June 1995. ISSN 0957-4271.

[36] Barry W. Peterson and Richard D. Boyle. Vestibulocollic reflexes. In Stephen M. Highstein, Richard R. Fay, and Arthur N. Popper, editors, *The Vestibular System*, number 19 in Springer Handbook of Auditory Research, pages 343–374. Springer New York, January 2004. ISBN 978-0-387-98314-1, 978-0-387-21567-9. URL `http://link.springer.com/chapter/10.1007/0-387-21567-0_8`.

[37] C. Walker, C. J. Vierck, and L. A. Ritz. Balance in the cat: role of the tail and effects of sacrocaudal transection. *Behavioural Brain Research*, 91(1-2):41–47, March 1998. ISSN 0166-4328.

[38] Craig R. White, Norman Day, Patrick J. Butler, and Graham R. Martin. Vision and foraging in cormorants: More like herons than hawks? *PLoS ONE*, 2(7), July 2007. ISSN 1932-6203. doi: 10.1371/journal.pone.0000639. URL `http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1919429/`.

[39] birdnote. Bird's eye view. `http://birdnote.org/`. consulted at 26-8-2014.

[40] Russell D Fernald. Evolving eyes. *The International journal of developmental biology*, 48(8-9):701–5, January 2004. ISSN 0214-6282. doi: 10.1387/ijdb.041888rf. URL `http://www.ncbi.nlm.nih.gov/pubmed/15558462`.

[41] César Miguel Rodrigues de Sousa. Preliminary study of the influence of vision in human balance. Project in automation engineering, University of Aveiro, June 2014.

[42] Y. Fukuoka, K. Tanaka, A Ishida, and H. Minamitani. Characteristics of visual feedback in postural control during standing. *IEEE Transactions on Rehabilitation Engineering*, 7(4):427–434, December 1999. ISSN 1063-6528. doi: 10.1109/86.808946.

[43] S. Ushida, K. Yoshimi, T. Okatani, and K. Deguchi. The importance of gaze control mechanism on vision-based motion control of a biped robot. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 4447–4452, Oct 2006. doi: 10.1109/IROS.2006.282079.

[44] Angela Faragasso, Giuseppe Oriolo, Antonio Paolillo, and Marilena Vendittelli. Vision-based corridor navigation for humanoid robots. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3190–3195. IEEE, 2013.

[45] Gary R Bradski. Computer vision face tracking for use in a perceptual user interface. 1998.

[46] Wyss Zurich. `http://www.wysszurich.uzh.ch/projects/zurich-eye/id/9/action/show/ctrl/Page/`. Consulted at 22-4-2016.

[47] Mark Maimone, Yang Cheng, and Larry Matthies. Two years of visual odometry on the mars exploration rovers. *Journal of Field Robotics*, 24(3):169–186, 2007.

[48] Manmohan Chandraker, Jongwoo Lim, and David Kriegman. Moving in stereo: Efficient structure and motion using lines. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1741–1748. IEEE, 2009.

[49] F Fraundorfer and D Scaramuzza. Visual odometry: Part i: The first 30 years and fundamentals. *Robotics & Automation Magazine, IEEE*, 18(4):80–92, 2011.

[50] Khalid Yousif, Alireza Bab-Hadiashar, and Reza Hoseinnezhad. An overview to visual odometry and visual slam: Applications to mobile robotics. *Intelligent Industrial Systems*, 1(4):289–311, 2015.

[51] Olivier Stasse, Andrew J Davison, Ramzi Sellaouti, and Kazuhito Yokoi. Real-time 3d slam for humanoid robot considering pattern generator information. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 348–355. IEEE, 2006.

[52] Alberto Pretto, Emanuele Menegatti, Maren Bennewitz, Wolfram Burgard, and Enrico Pagello. A visual odometry framework robust to motion blur. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 2250–2257. IEEE, 2009.

[53] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 47(1-3):7–42, 2002.

[54] SKF. *Needle roller bearings*. SKF Group, March 2010.

[55] Hitec. hitec-hsr-5498sg-digital-servo. `http://www.robotshop.com/media/files/pdf/hitec-hsr-5498sg-digital-servo-specsheet.pdf`. consulted at 17-9-2014.

[56] EmÃlio Geraldo Estrelinha. Tele-operation of a humanoid robot using haptics and load sensors. Master's thesis, University of Aveiro, 2013.

[57] Firefly MV. http://www.ptgrey.com/support/downloads/10116, July 2011. URL http://www.ptgrey.com/support/downloads/10116.

[58] ROS. http://wiki.ros.org/camera_calibration/Tutorials/MonocularCalibration, March 2015. URL http://wiki.ros.org/camera_calibration/Tutorials/MonocularCalibration.

[59] FANUC CORPORATION. http://www.euromach.gr/xmsAssets/Image/Demo/Products/RoboticSystems/RobotArms/LRMate100iB/s/s_1.jpg?1190795426, July 2015. URL http://www.euromach.gr/xmsAssets/Image/Demo/Products/RoboticSystems/RobotArms/LRMate100iB/s/s_1.jpg?1190795426.

[60] FANUC CORPORATION. Fanuc m-6ib. https://www.robots.com/fanuc/m-6ib-6s, 2015. consulted at 23-9-2015.

[61] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully B. Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. ROS: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[62] Miguel Armando Riem de Oliveira. *Automatic Information and Safety Systems for Driving Assistance*. PhD thesis, University of Aveiro, 2013.

[63] ROS. Bags. http://wiki.ros.org/Bags, 2015. consulted at 23-9-2015.

[64] Ivan Culjak, David Abram, Tomislav Pribanic, Hrvoje Dzapo, and Mario Cifrek. A brief introduction to opencv. In *MIPRO, 2012 Proceedings of the 35th International Convention*, pages 1725–1730. IEEE, 2012.

[65] Gary Bradski and Adrian Kaehler. *Learning OpenCV*. Reilly Media, Inc, 2008.

[66] Bernard Espiau, François Chaumette, and Patrick Rives. A new approach to visual servoing in robotics. *Robotics and Automation, IEEE Transactions on*, 8 (3):313–326, 1992.

[67] Lagadic project INRIA Rennes-Bretagne Atlantique. Visp tracking methods overview. 2010.

[68] Srikanth Lukka Pulipati Annapurna, Sriraman Kothuri. Digit recognition using freeman chain code. *International Journal of Application or Innovation in Engineering & Management (IJAIEM)*, 2(8):362–365, 2013. URL http://www.ijaiem.org/volume2issue8/IJAIEM-2013-08-31-081.pdf.

[69] Nor Amizam Jusoh and Jasni Mohamad Zain. Application of freeman chain codes: An alternative recognition technique for malaysian car plates. *arXiv preprint arXiv:1101.1602*, 2011.

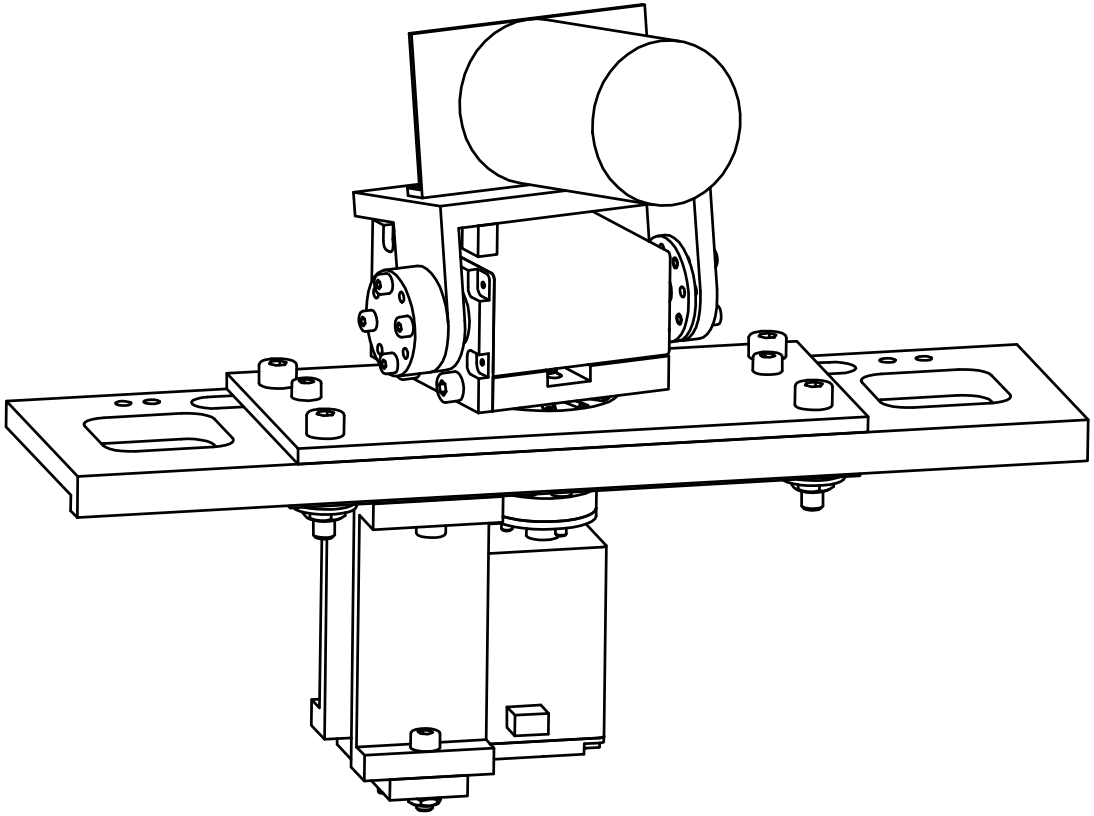
[70] P. McKerrow. *Introduction to Robotics*. ACM Press frontier series. Addison-Wesley Publishing Company, 1991. ISBN 9780201182408. URL `https://books.google.pt/books?id=6jVSAAAAMAAJ`.

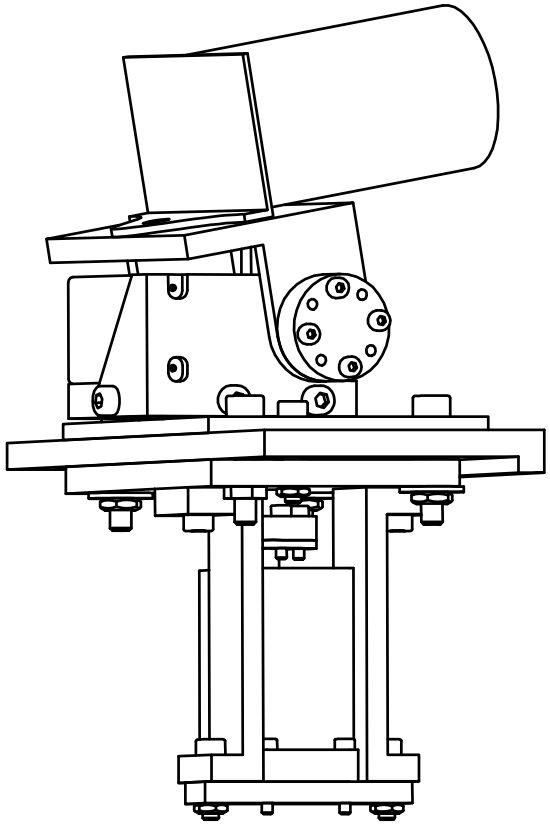
[71] Adorama. Flashpoint zerograv 2-axis digital gyro stabilizer. `http://www.adorama.com/FPSTABG.html`, 2013. consulted at 31-12-2015.

[72] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, 2014.

Appendix A

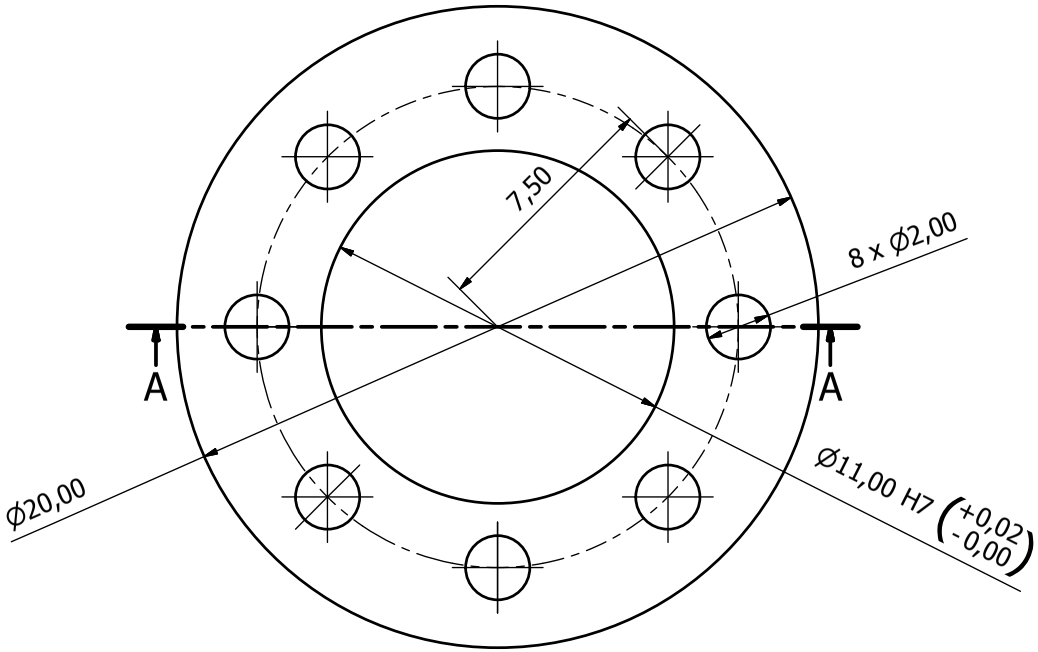
Mechanical Drawings

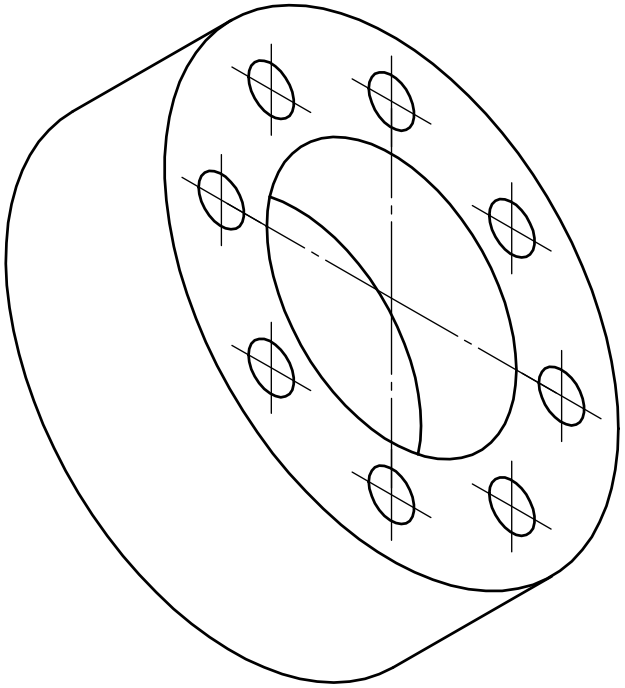




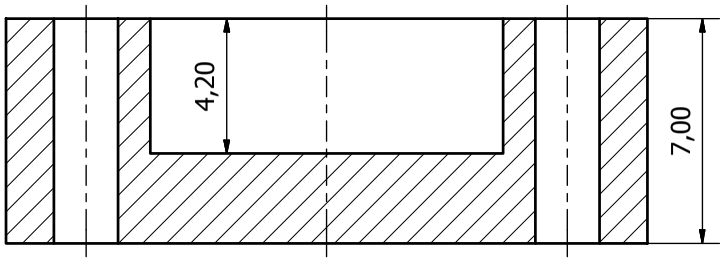
Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	draw_mont_base_v0	Edição	Folha 1 / 1





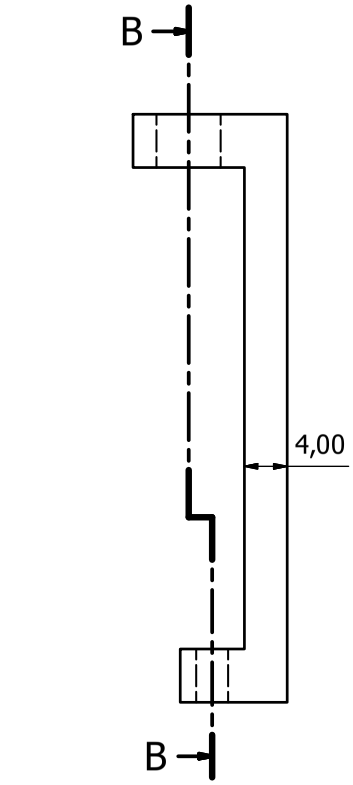
A-A (6 : 1)

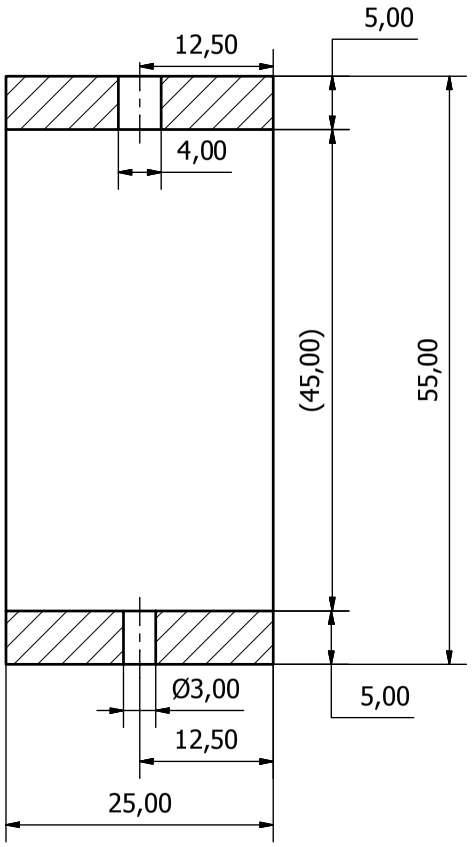


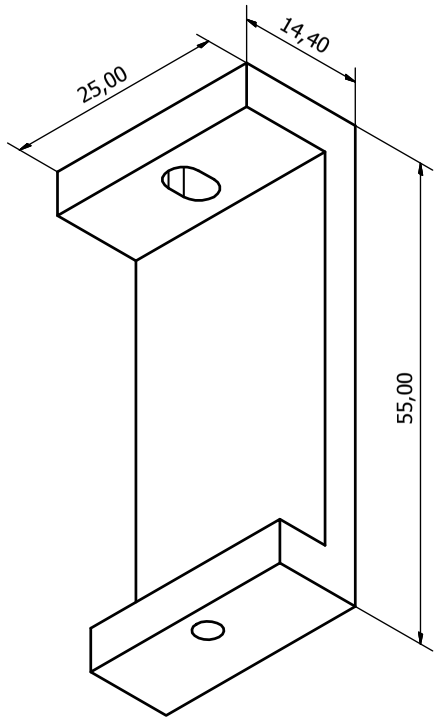
Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

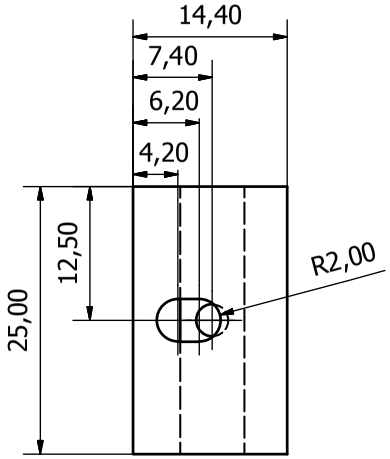
Projeto Humanoid UA- Prof. Vitor Santos e Filipe Silva	1 Peça		
	draw_caixa_rolamento_v1	Edição	Folha 1 / 1

B-B (2 : 1)





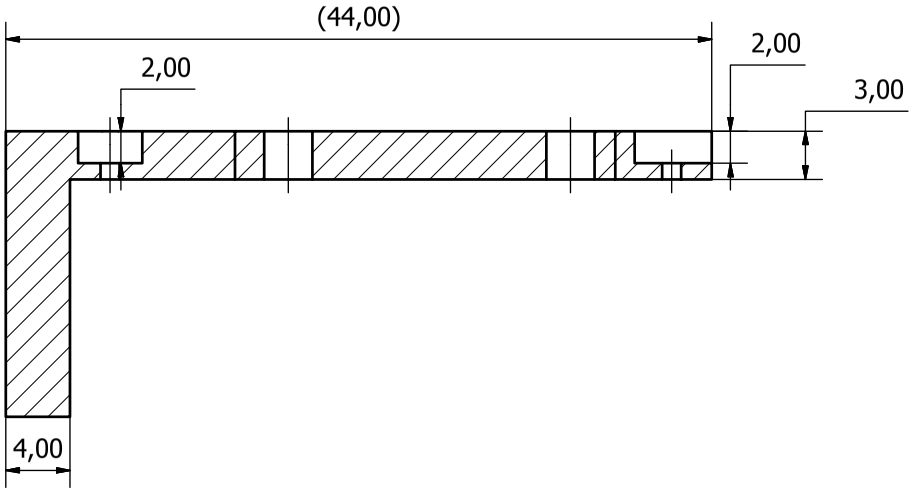


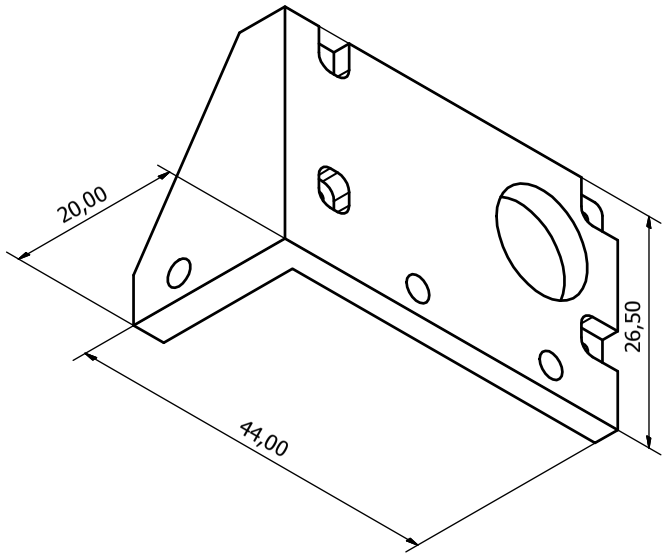


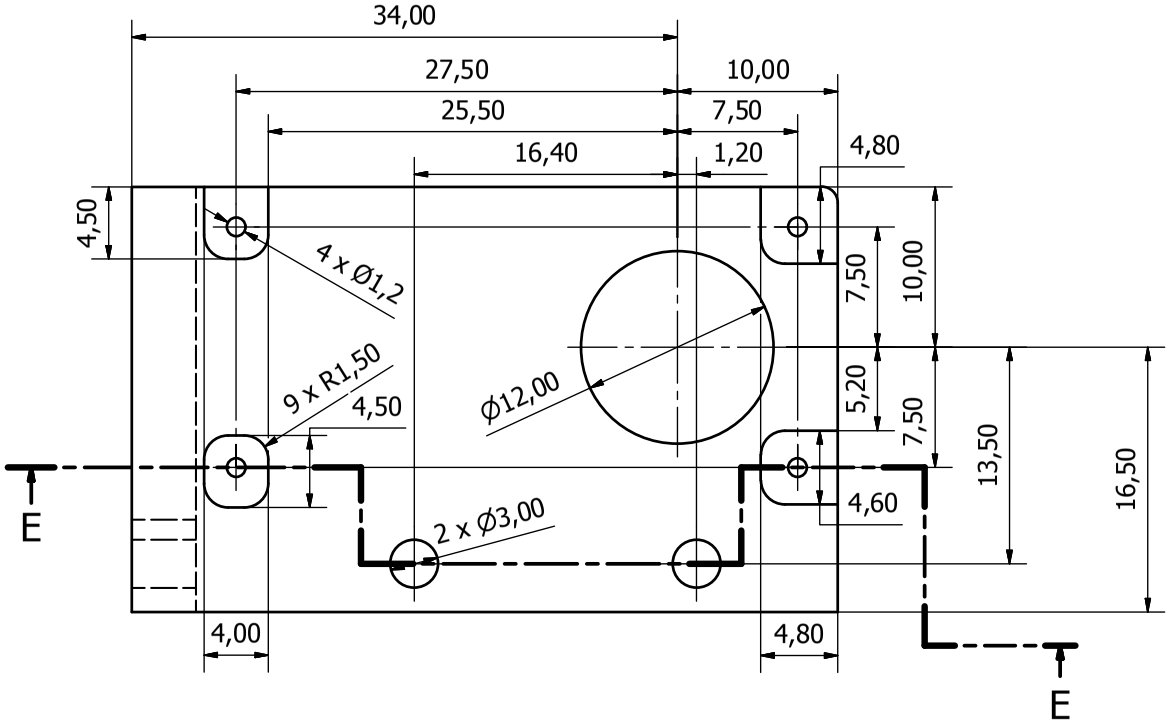
Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

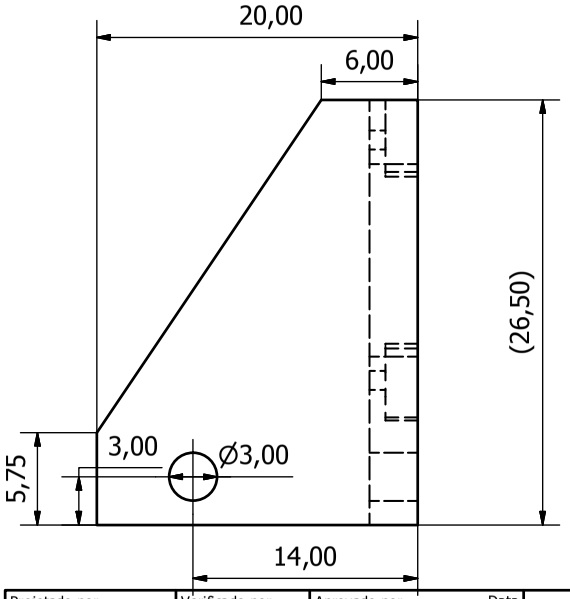
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	2 PEÇAS		
	draw_suporte_servo_baixo_v2	Edição	Folha 1 / 1

E-E (3 : 1)



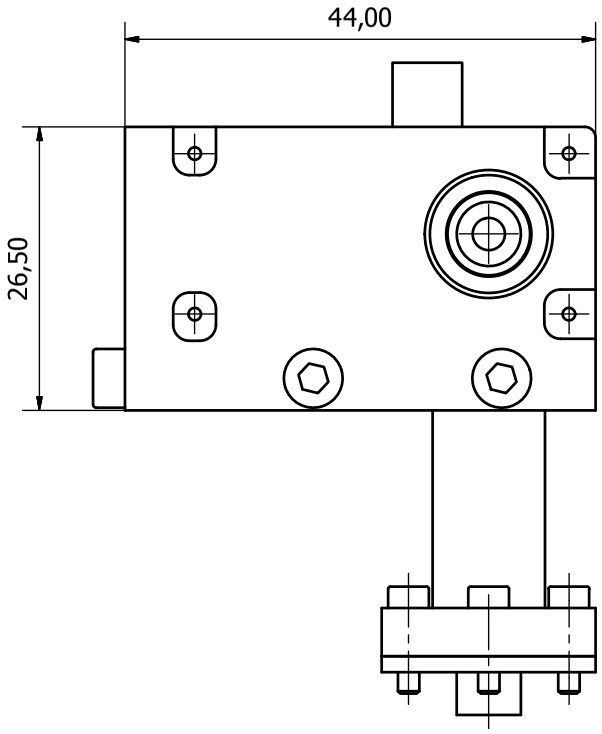


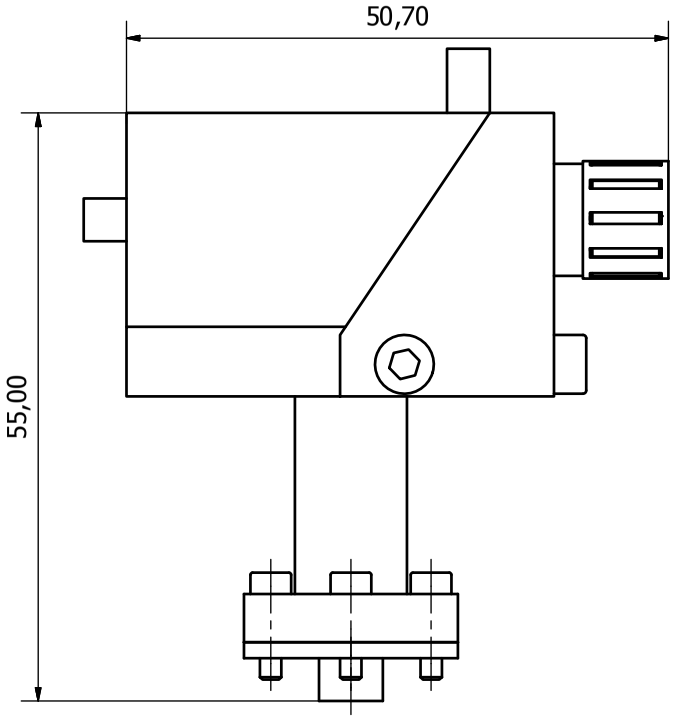


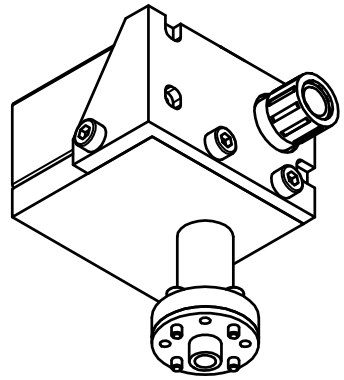


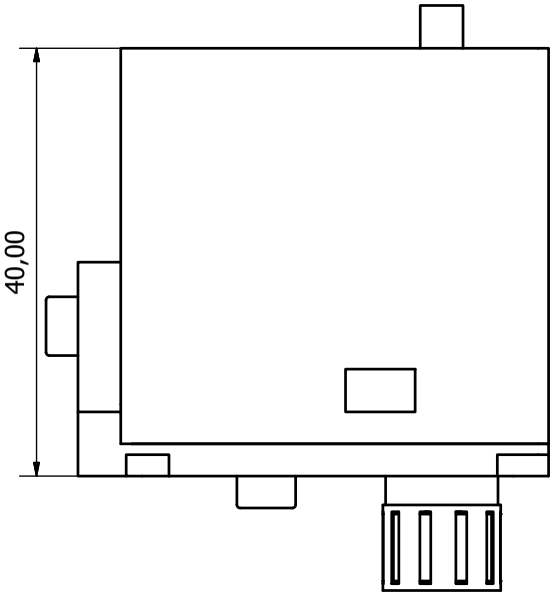
Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça		
	draw_chapa_servo2_v2	Edição	Folha 1 / 1





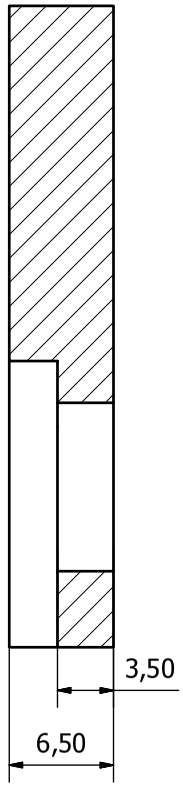


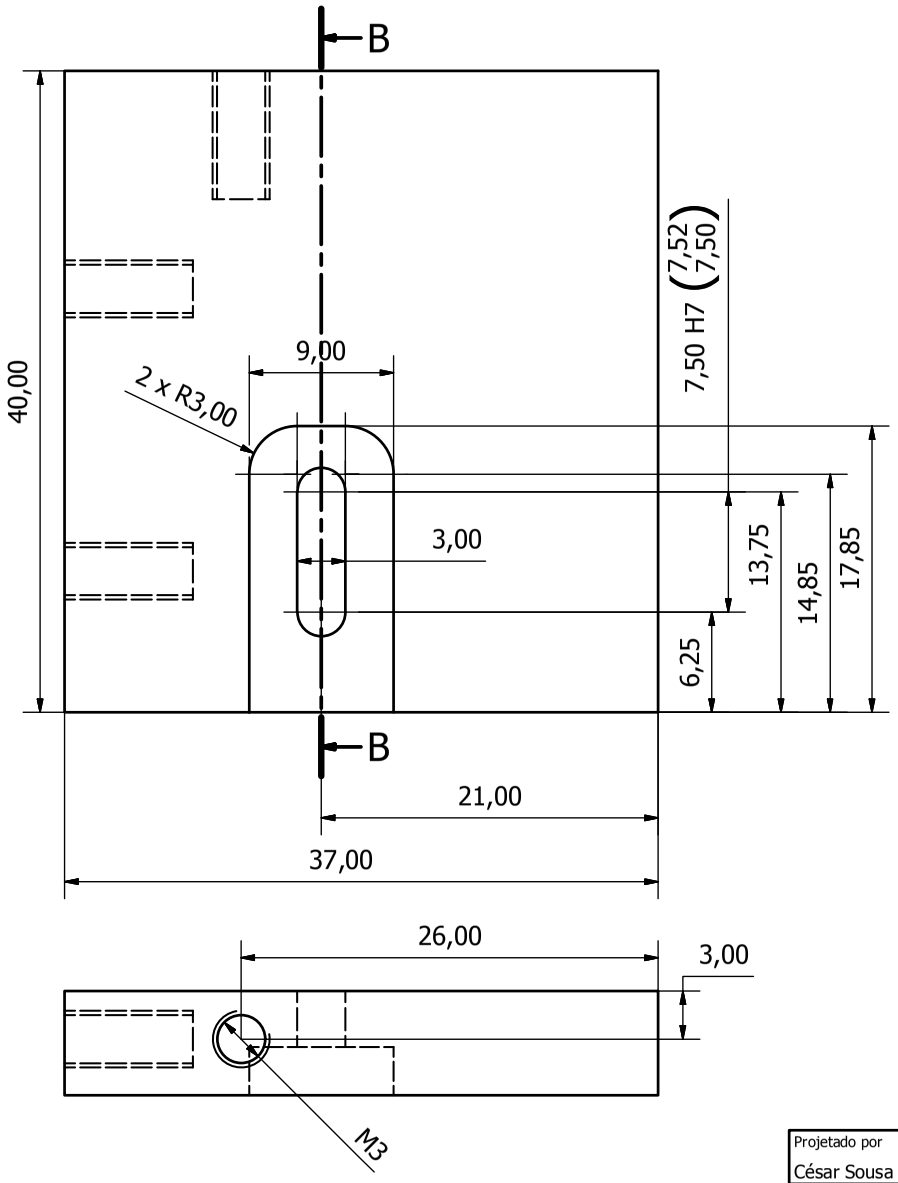


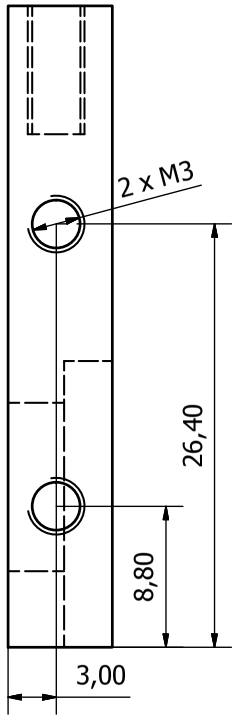
Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

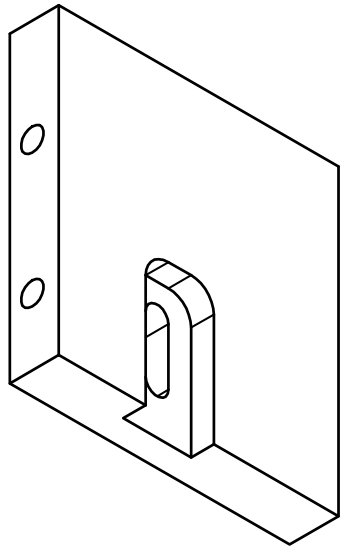
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça		
	draw_mont_meio-1	Edição	Folha 1 / 1

B-B (3 : 1)





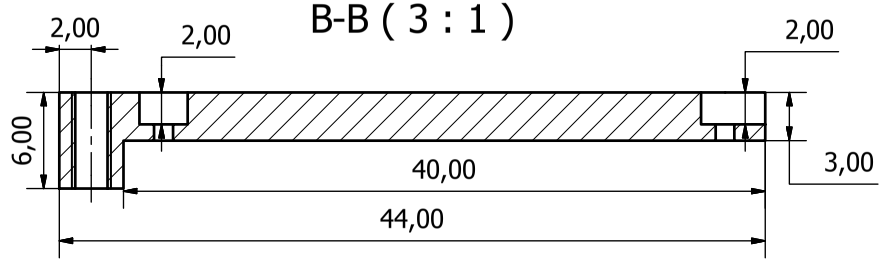


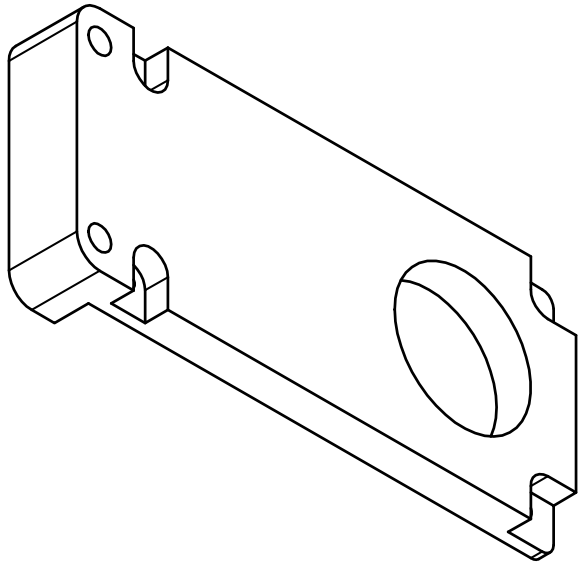


Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

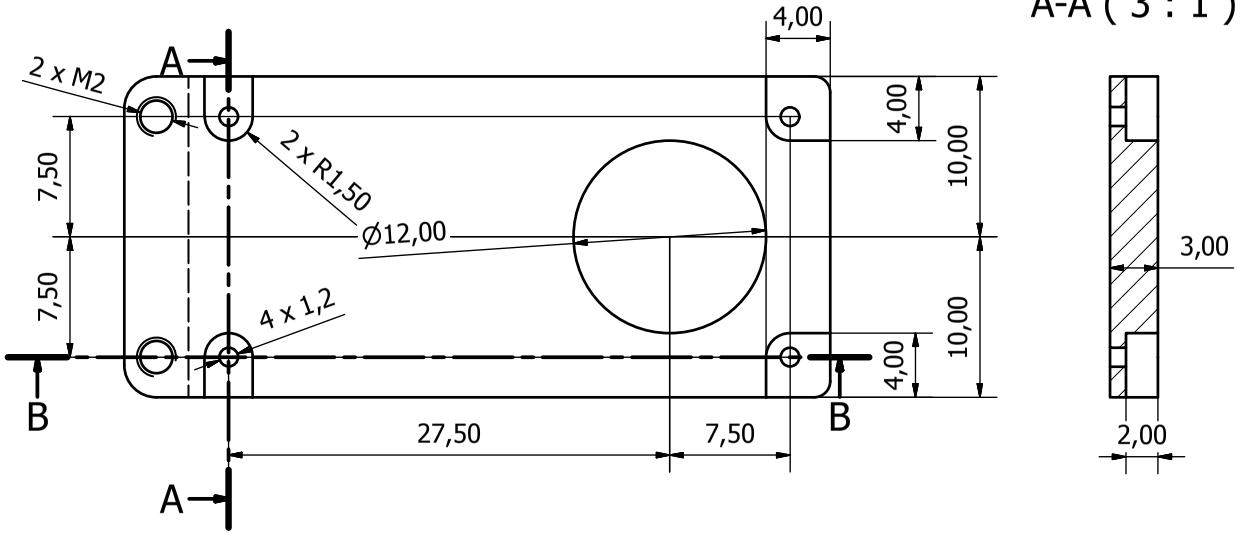
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça		
	draw_suporte_servo_v5	Edição	Folha 1 / 1

B-B (3 : 1)



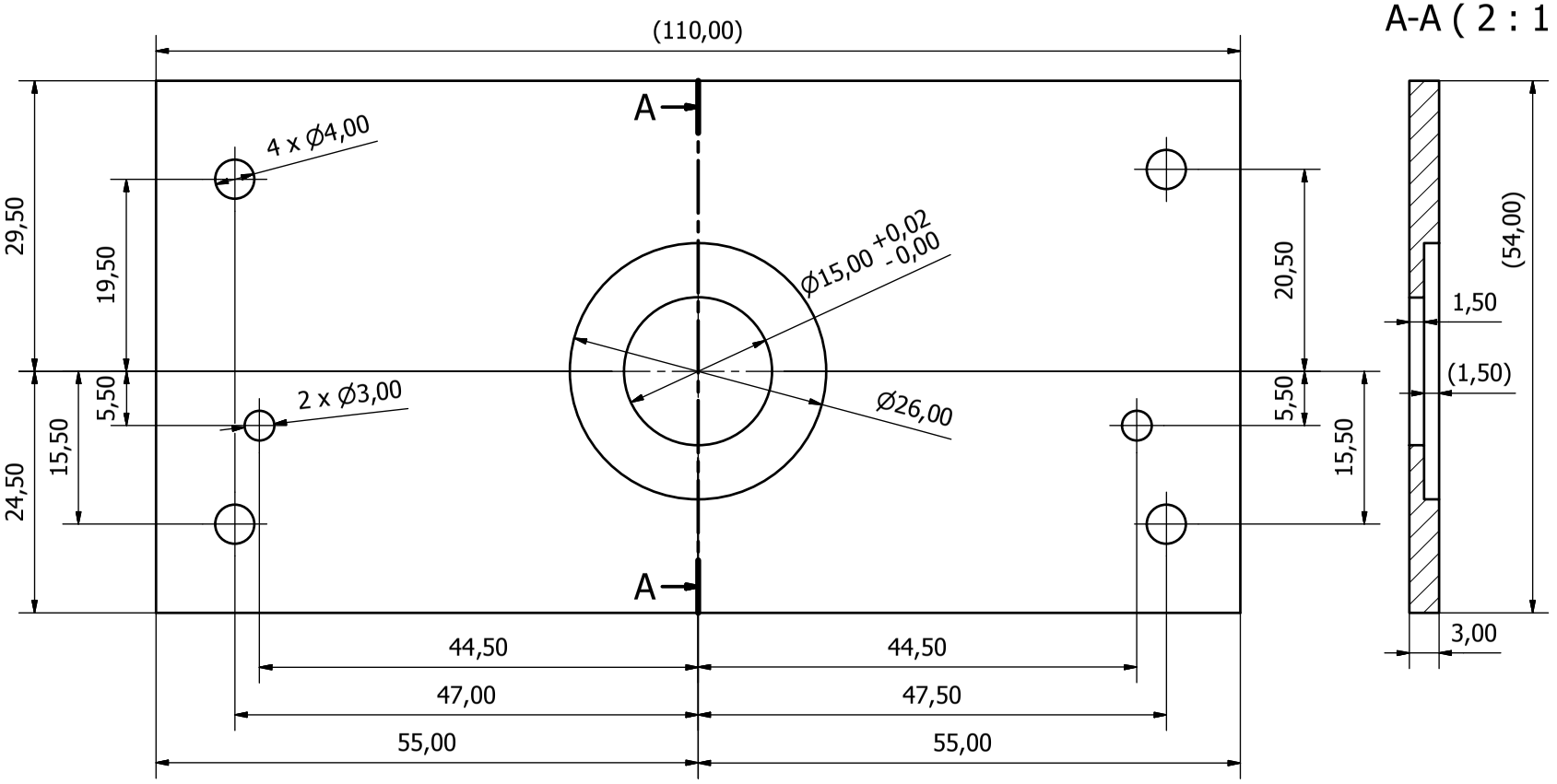


A-A (3 : 1)

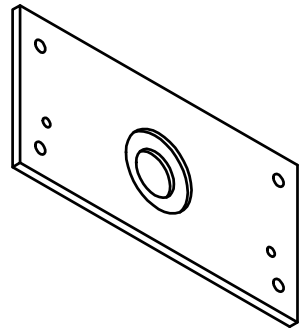


Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça		
	draw_chapa_servo_v2	Edição	Folha 1 / 1

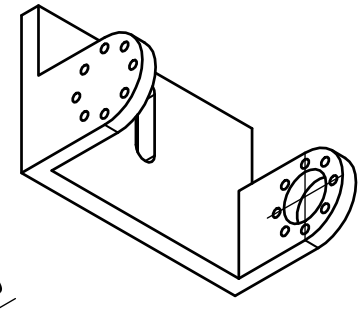


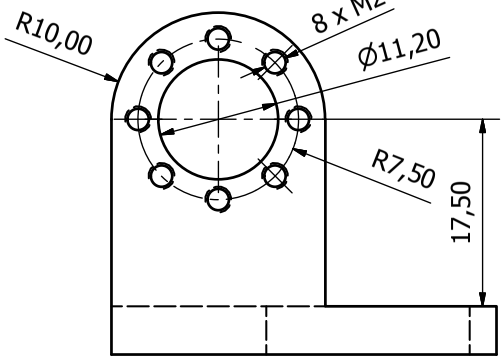
A-A (2 : 1)

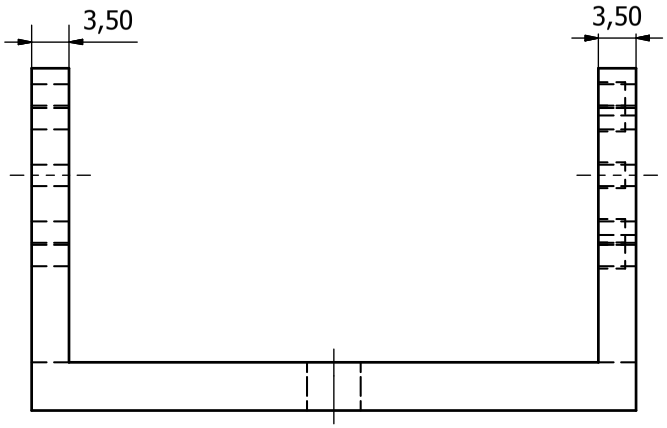


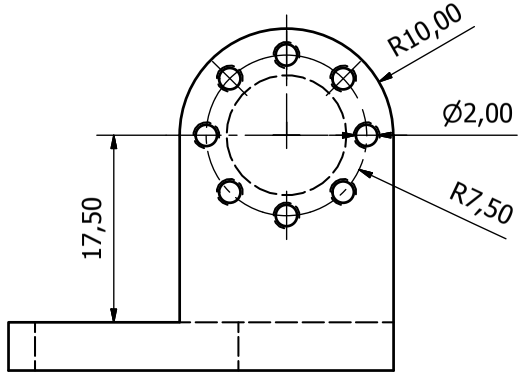
Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

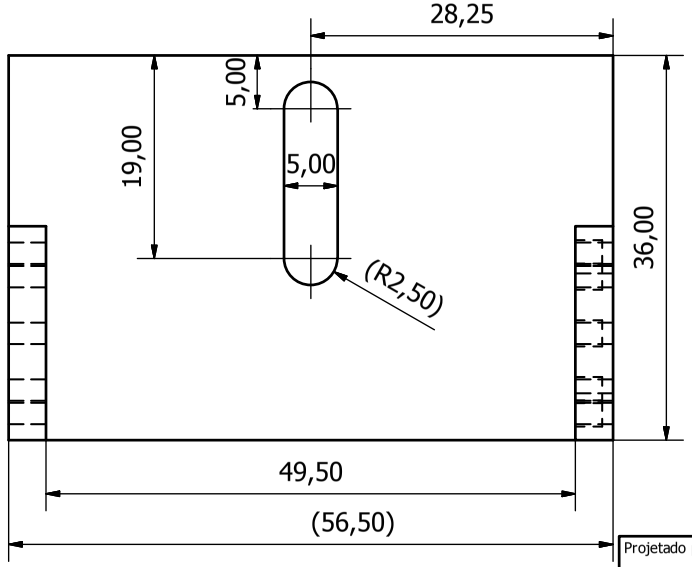
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça		
	draw_suporte_rolamento-1	Edição	Folha 1 / 1







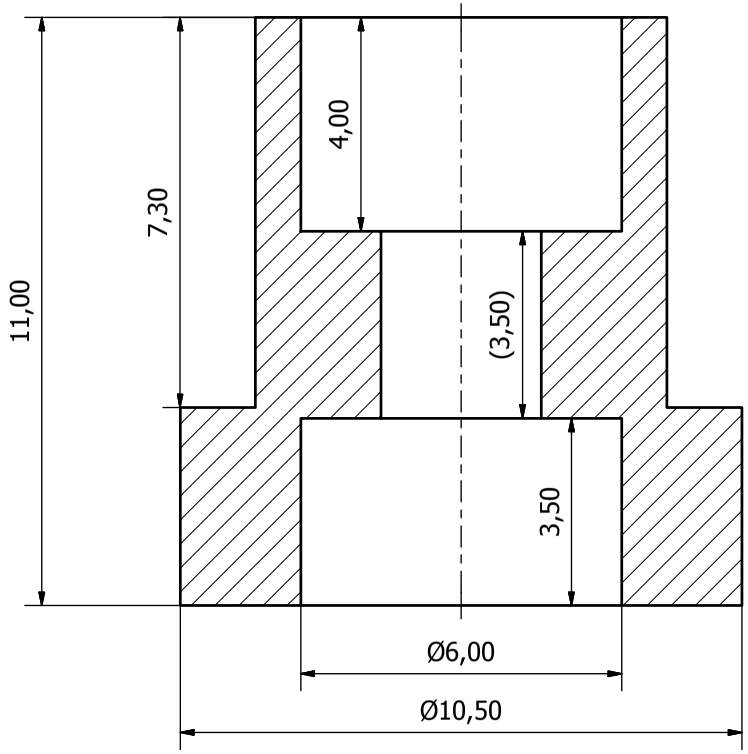


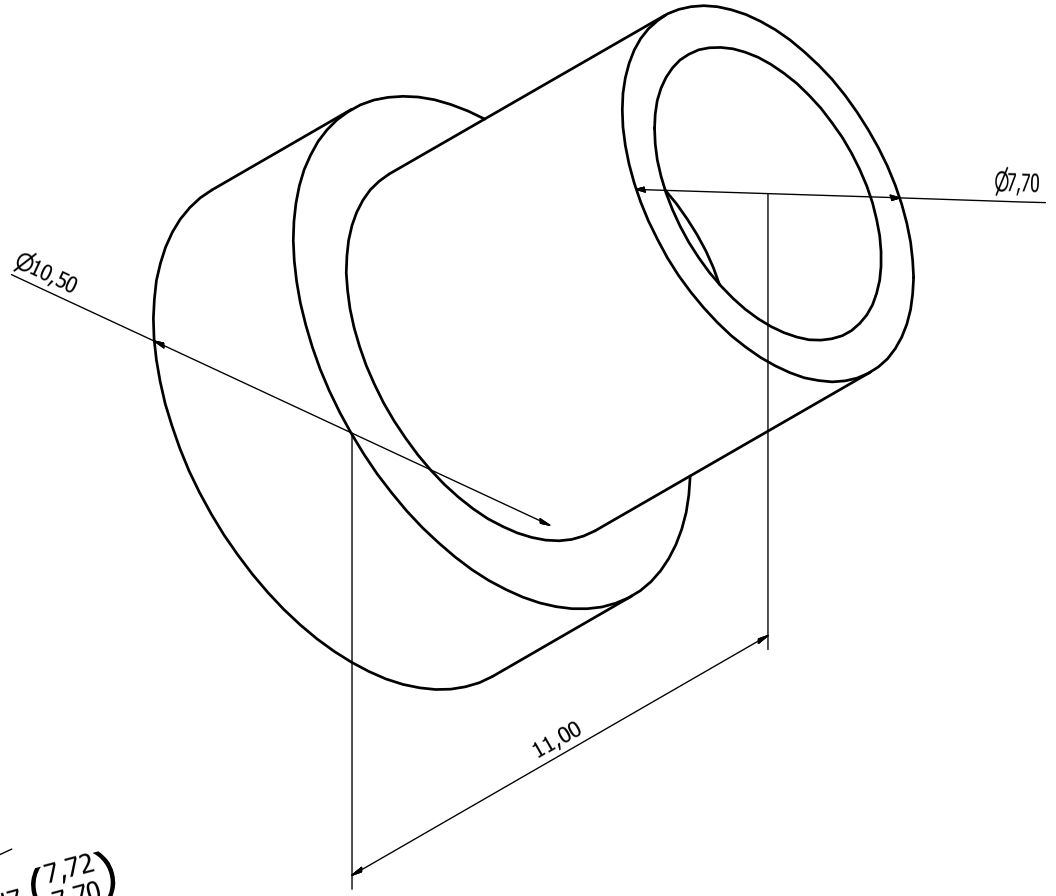


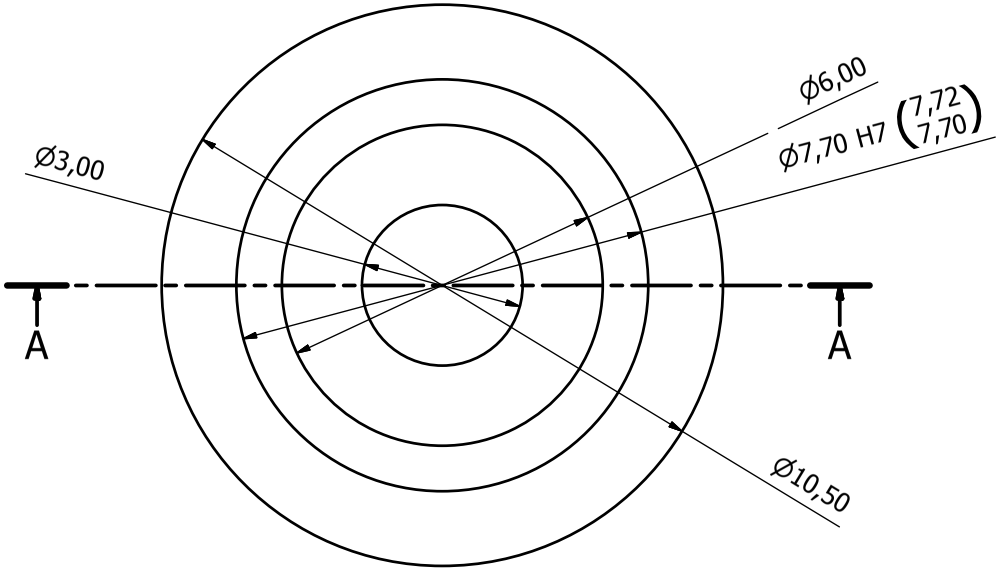
Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça		
	drwa_ligacao_servo_camara2_v2	Edição	Folha 1 / 1

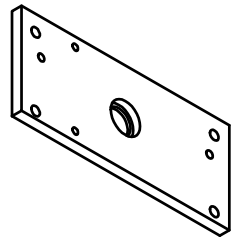
A-A (10 : 1)

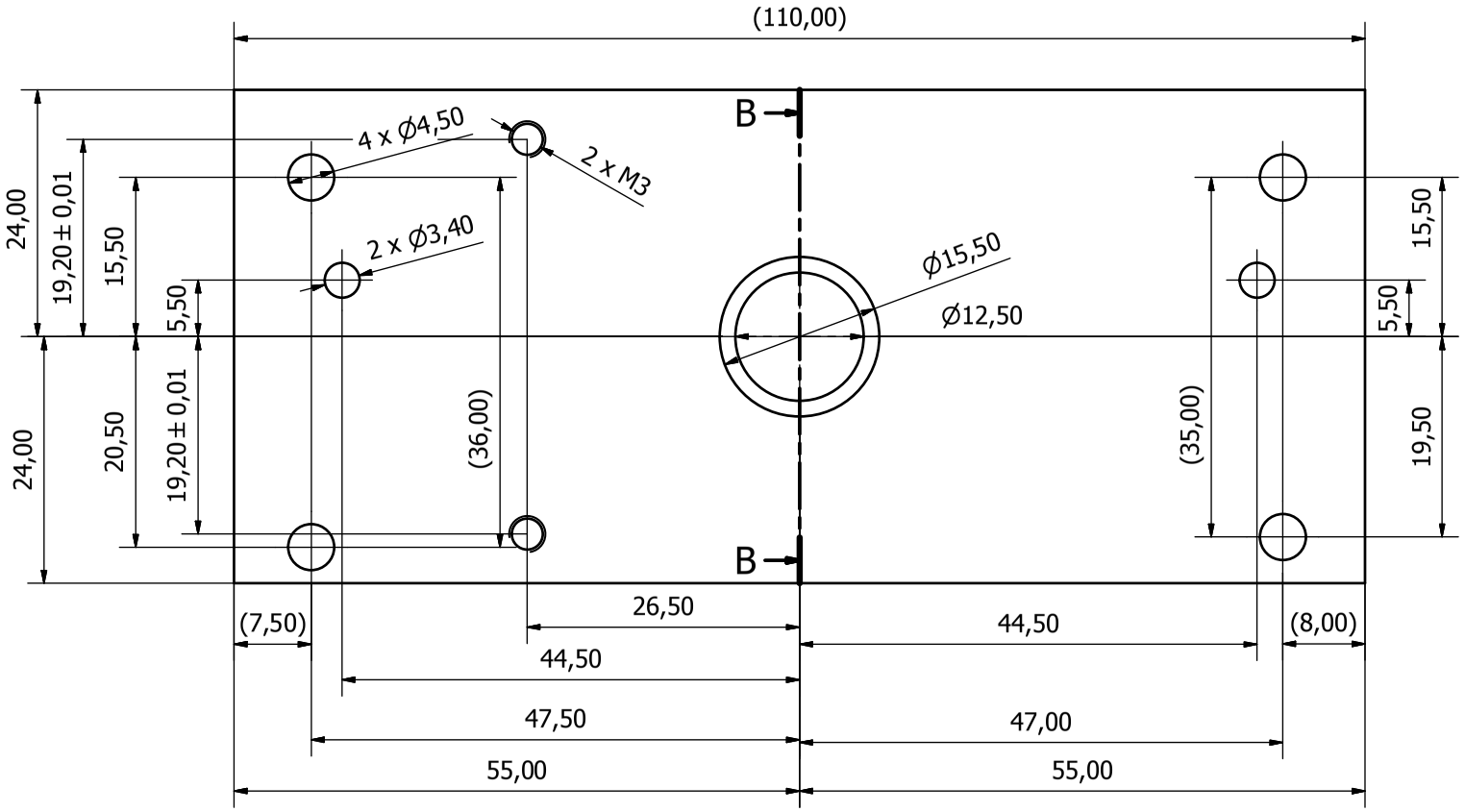




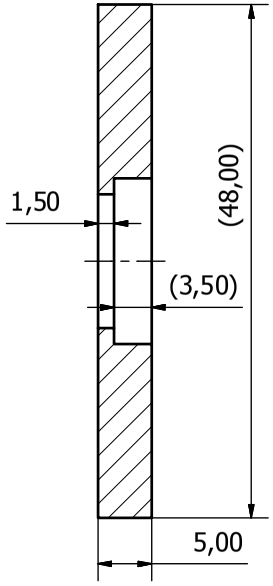


Projetado por César Sousa 49649	Verificado por	Aprovado por	Data		Data 02-12-2013	
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva				1 peça		
				draw_guia_servo_v0	Edição	Folha 1 / 1



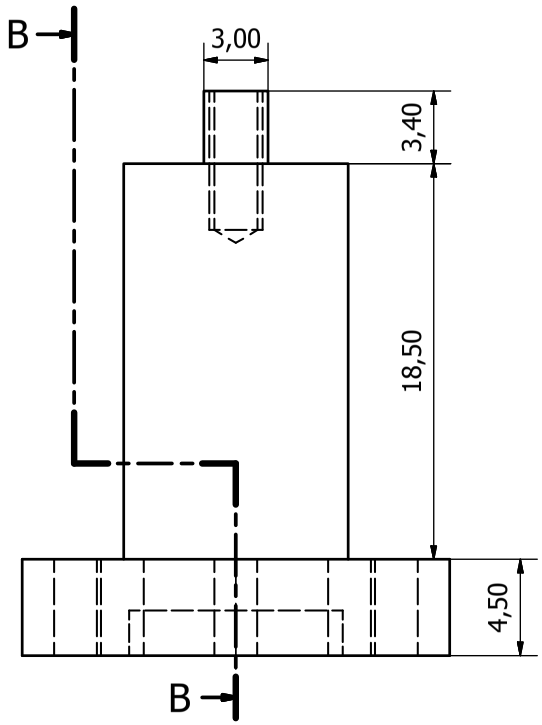


B-B (2 : 1)

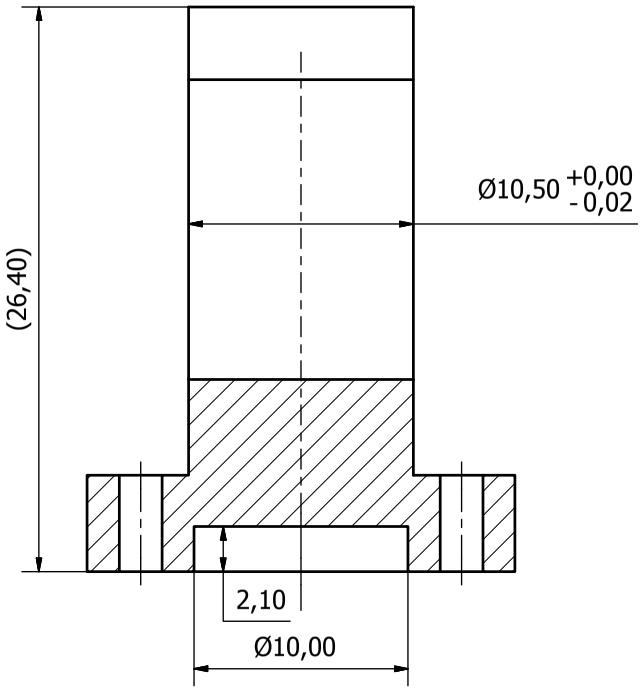


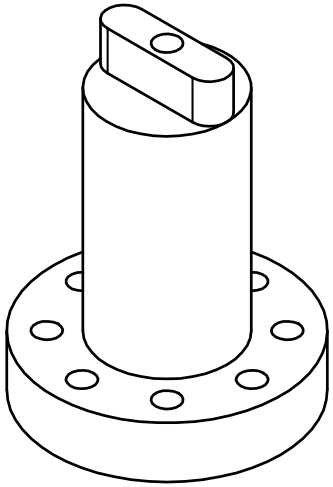
Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

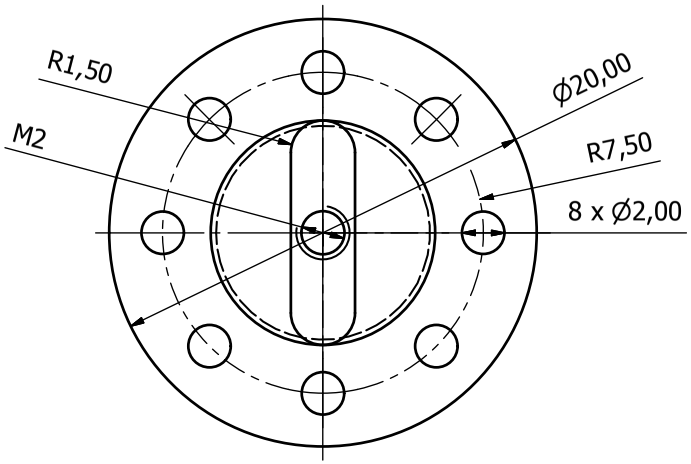
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça		
	draw_suporte_rolamento_guia_v0	Edição	Folha 1 / 1



B-B (4 : 1)

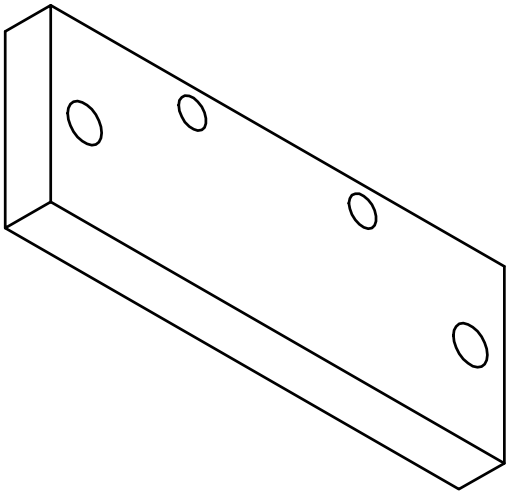




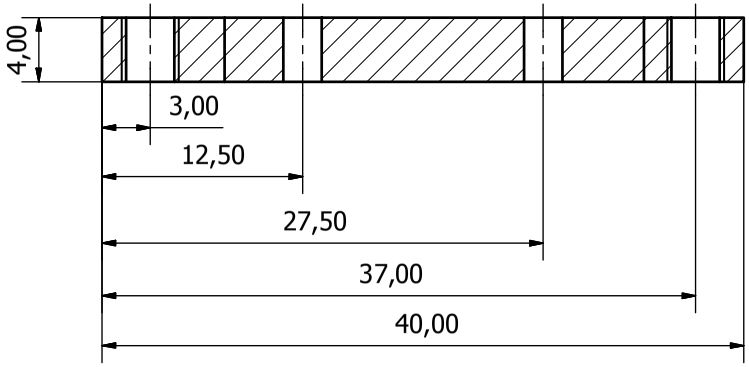


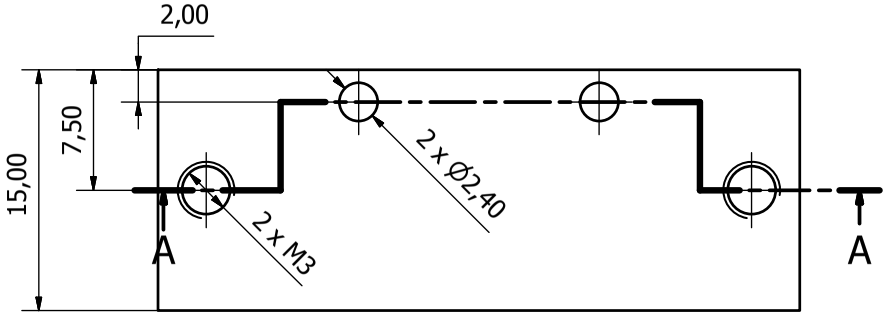
Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça		
	draw_ligacao_entre_servos_v4	Edição	Folha 1 / 1



A-A (3 : 1)





Projetado por	Verificado por	Aprovado por	Data		Data	
César Sousa 49649					02-12-2013	

Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça		
	draw_suporte_servo_baixo2_v2	Edição	Folha 1 / 1