Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2016

César Miguel Rodrigues de Sousa

Controlo Postural de Robô Humanóide Baseado em Realimentação Visual

Visual Feedback to Assist Humanoid Balance

Universidade de Aveiro
2016

Departamento de Eletrónica,
Telecomunicações e Informática

César Miguel
Rodrigues de Sousa

Controlo Postural de Robô Humanóide Baseado em Realimentação Visual

Visual Feedback to Assist Humanoid Balance

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 João Antunes da Silva professor associado da Faculdade de Engenharia da Universidade do Porto
vogais / examiners committee	Prof. Doutor João Antunes da Silva professor associado da Faculdade de Engenharia da Universidade do Porto
	Prof. Doutor João Antunes da Silva professor associado da Faculdade de Engenharia da Universidade do Porto
	Prof. Doutor João Antunes da Silva professor associado da Faculdade de Engenharia da Universidade do Porto
	Prof. Doutor João Antunes da Silva professor associado da Faculdade de Engenharia da Universidade do Porto
	Prof. Doutor João Antunes da Silva professor associado da Faculdade de Engenharia da Universidade do Porto

agradecimentos / acknowledgements

Em Primeiro lugar 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 o seu sucesso.
Ao Eng. Festas pela ajuda na concretização do novo pescoço do humanoide e à empresa Motofil por disponibilizar o robot que permiu realizar a parte experimental da dissertação.

À ESSUA pela disponibilização do equipamento VICON, nomeadamente o professor António Amaro, reponsável pelo equipamento e o 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 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 ùltimo, 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 postural, *Feedback* visual, Seguimento, Robótica, Humanoides, Equilíbrio e ViSP.

Resumo

O equilíbrio é uma das componentes fundamentais dos robôs humanoides. Assumindo a existência de características fixas no cenário envolvente, este trabalho tem como tese que a visão poder ter um papel importante no equilíbrio de robôs humanoides assim como tem nos seres humanos. O Projeto Humanoide da Universidade de Aveiro (PHUA) é usado neste trabalho como base para desenvolver a proposição desta dissertação e todos os componentes mecânicos do pescoço do PHUA foram reconstruidos e melhorados para assegurar uma boa base. Foram desenvolvidos algoritmos de processamento de imagem e seguimento visual para encontrar e seguir na imagem, um alvo fixo, com o intuito de obter *feedback* visual para o sistema de seguimento do pescoço. Desenvolveu-se também um algoritmo de controlo de seguimento com o intuito de seguir o alvo baseado no *feedback* visual. A informação da posição do pescoço pode ser posteriormente integrada com a rede sensorial do humanoide com o intuito de molhar o equilíbrio do robô. O desenvolvimento do *software* foi baseado numa plataforma modular que permite a criação de vários modos de funcionamento independentes. Para simular os movimento do humanoide com a intenção de testar o sistema de seguimento desenvolvido foi utilizado um robô antropomórfico industrial. Os resultados dos testes demonstraram que os algoritmos de visão por computador tiveram uma boa performance face às especificações e o controlo de seguimento baseado em velocidade é o melhor para obter um simples e bom sistema de seguimento visual para o humanoides.

Keywords

Postural control, Visual feedback, Tracking, Robotics, Humanoids, Balance and ViSP.

Abstract

Balance is one of the key elements of the humanoid robots. Assuming the existence of fixed characteristics on the scene, this work addresses the thesis that vision may play a major role in humanoids balance 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. An image processing and tracking algorithms were developed to find and track an fixed target on an image, aiming to give visual feedback for the neck's tracking system. 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. 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 system. 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 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
ZMP	Zero Moment Point

Chapter 1

Introduction

Humanoid robots will be the revolution of the XXI century [31].

To make a humanlike 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, 46]. Given the fast development of the technology, today humanoids have become one of the main focus on robotics research and many efforts have been made by many companies and research groups around the world in order to make that dream come true [3].

Since the 70s, many famous humanoid robots have emerged in the society, such as ASIMO[1] made by the company Honda, NAO by Aldebaran Robotics or the NASA's[2] Robonaut. All of them are represented bellow in the Figure 1.1 [5, 25, 27, 31, 29, 56].

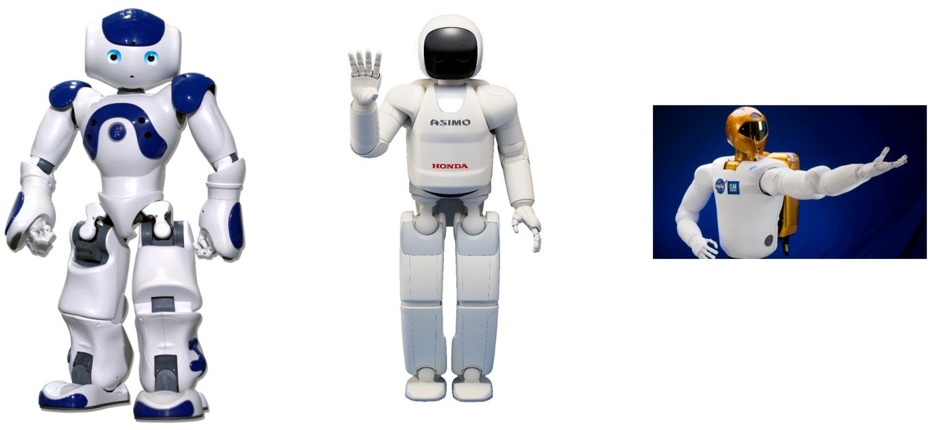


Figure 1.1: Most famous humanoid robots a)ASIMO by Honda [29], b) NAO by Aldebaran Robotics [47] and c) Robonaut by NASA [36]

Due to its biped support, some of the most important and challenging concerns in

[1]Advanced Step in Innovative Mobility.

[2]National Aeronautics and Space Administration.

this area are the standing balance, walking and consequently, intra-dynamics control [58, 39]. Humanoids need sophisticated abilities to adapt to unknown situations and environments, therefore, as more information the humanoids obtain about themselves and the environment around, the better they will perform [57].

Some of the key points that guarantee a good postural behavior in the robot are the active alignment of the trunk and the head in relation to gravity, the support surface, the visual environment and internal references. The maintenance of a stable posture has been essentially associated, from the robotics point of view, with the control of COP (Center Of Pressure) on the feet or/and the Zero Moment Point (ZMP)[3], which is a well-known indicator of dynamic equilibrium but, has a reduced use of vestibular signals[4] or vision [38, 37].

Vision sensors are common in most humanoid robots, intending to detect features or structures on the scene in order to perform classification and object recognition or even plan locomotion routes[54].

The motivation of this Dissertation relies on the hypothesis that **assuming the existence of fixed characteristics on the scene, vision may play a major role in humanoids balance such as it is on humans [17, 52, 26, 58].**

This idea is not new. Fukuoka, Y. *et al.* introduced this concept in 1999, nevertheless, without success, concluding that visual feedback has a large time delay [23]. Afterwards, in 2006, Ushida S. *et al.* emphasized the need of visual feedback combining gaze control with image processing. Given the evolution of technology over the last years enabling a better performance of the 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 [59, 58]:

> "The complementary or concurrent utilization of the vision with internal sensors provides more reliable control structure of biped walking systems in the future." [38]

This findings needs more investigation. Faragasso A. *et al.* affirm that it is necessary to boost the usage of exteroceptive sensing capabilities given the complexity of these robotic systems [20]. The author proposes to further study and start the implementation of this approach on a ongoing larger project called Humanoid Project of the University of Aveiro (PHUA) where the research activities are conducted on.

[3]Where the total of horizontal inertia and gravity forces equals zero [64].

[4]The sensory system that provides the leading contribution about movement and sense of balance [64].

1.1 PHUA

With the intention to build a low cost platform for students to do research on a such interesting topic as robotics, allowing them to extend their knowledge in a wide area of technologies including mechanics and electronics. The Department of Mechanical Engineering (DEM) and the Department of Electronics, Telecommunications and Informatics (DETI) of the University of Aveiro has started at the Laboratory of Automation and Robotics (LAR) the - PHUA -, where this Dissertation has emerged [53].

The PHUA, illustrated in the Figure 1.2, has begun in 2004 with its mechanical design. Over the years several master students have been making their dissertations on this project reaching at the moment an humanoid platform with 667 millimeters of height, 6 kilograms and 27 Degrees Of Freedom (DOF) [24].

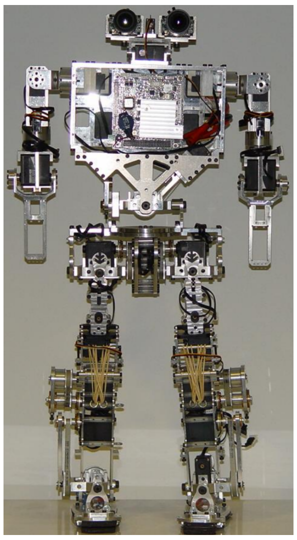


Figure 1.2: Humanoid Project of the University of Aveiro [32].

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 [13]. The PHUA also possesses 8 pressure sensors under the feet and a fire-wire camera placed on the head [45].

In respect of the PHUA's balance, to approach a concept for robot learning by demonstration, Cruz P.[13] 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 $7Hz$ [45].

This is an evolving project in which the author aims to give another reliable source of information to the PHUA's ability to balance, the vision sense.

1.2 Objectives

Given the presented motivation and the context of PHUA, the objectives for this Dissertation are:

- To study the main references in the state of the art concerning the visual-feedback influence in humans and how the scientific community introduced this concept in humanoid robotics;
- To analyze the PHUA's platform in mechanical, electrical and software point of view in order to layout its current state and understand how the robot could track the targets for the visual-feedback.
- To extract the correct characteristics from the scene that would serve as visual references for the robot;
- To align the head with those features;
- To propose an approach on how should the visual-feedback influence be implemented on PHUA's platform and therefore, on humanoid robots in general.

1.3 Dissertation Outline

Following the draft of the objectives, this Dissertation is divided into six chapters. The present Chapter is an introduction of its context and aims to address the question propose that the author. The second Chapter aims to give a review of state of the art about the influence of vision in humans balance and how this concept has been, or not, implemented into humanoid robots.

The third Chapter provides a review over the PHUA's suitability to implement the proposed objectives and information about the adaptation made in PHUA's mechanical structure. The Chapter four brings forward how the image acquisition was performed and how the image processing was made.

Chapter five describes how the infrastructure was tested as well as the subsequent results. Finally, Chapter six presents the conclusions and the open issues, to be addressed in future works.

Chapter 2

State of the Art

This Chapter conducts a literature review about the process of balancing humans as well as humanoids, given the fact that the PHUA is an human-like robot. Following the line of the objectives of this Dissertation, an overview about the role of the vision sense on this process in the physiology and anatomy point of view is made and summarized the state of the art related to the implementation of the visual-feedback in humanoid robots concept.

2.1 Balance

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 the heath point of view:

> "(...) act of maintaining, achieving or restoring a state of balance during any posture or activity." [41]

Balance is also referred as an amount of *postural sway*, which in practice are slight movements made in order to maintain a balanced position. Balance in humans is associated with the input from multiple sensory systems including the *vestibular*[1], *somatosensory*[2], and *visual systems* working together.

The *cerebellum* in human beings is the organ that coordinates all the 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 [60]. In fact, as Jones G. reffered , conscious control can be tragic [17].

[1]Sense of spatial orientation [64].

[2]Complex sensory system made up of a number of different receptors [64].

The *brain stem* contains centers for visual and auditory reflexes, which enables to initiate the movement [60]. Balance is also context dependent and there are several levels of nervous system that must be considered to account its complexity [17]. The Figure 2.1 illustrates the position of the *cerebellum*, *brain stem* and the *cerebral cortex* in a human head.

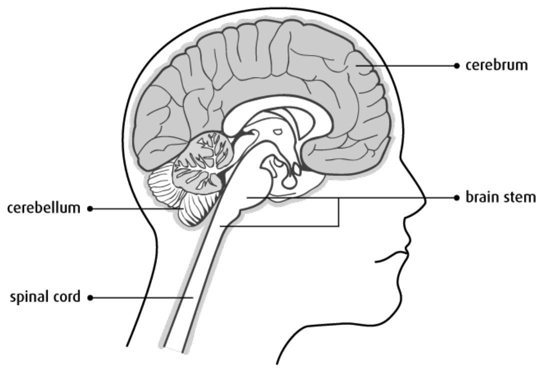


Figure 2.1: Position of cerebellum and cortex in the brain (modified from [9]).

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 parts of the body relative to one another [8]. The Figure 2.2 represents a diagram of the interactions between the main parts of the cerebrum that are responsible to maintain a postural sway.

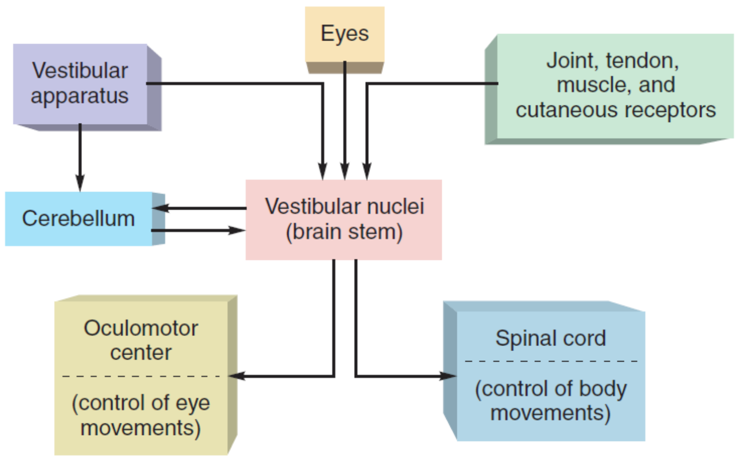

Eyes
Vestibular apparatus
Joint, tendon, muscle, and cutaneous receptors
Cerebellum
Vestibular nuclei (brain stem)
Oculomotor center
(control of eye movements)
Spinal cord
(control of body movements)

Figure 2.2: Neural pathways involved in the maintenance of equilibrium and balance in humans (adapted from [52]).

The study of balance in humans undoubtedly demands a much deeper review, that goes out of the scope of this dissertation. In the figure 2.2 it is verifiable that the vision system, through the eyes, has influence in humans balance.

2.1.1 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 the 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 [16]. 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[3], a force platform to determine the force made by the feet and a head-mounted camera to infer what is the available visual information for the subject at every given moment.

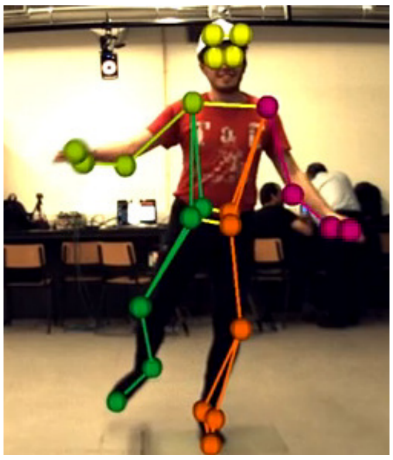
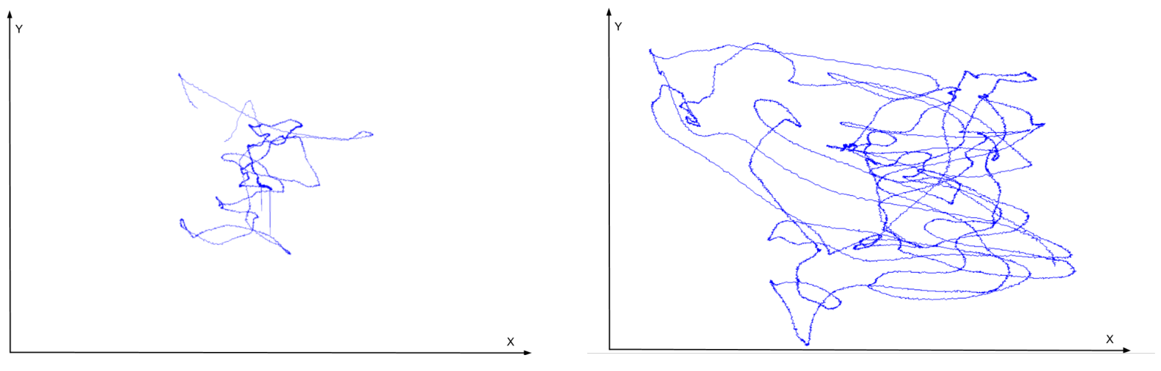


Figure 2.3: Empirical experiment at ESSUA department at the VICON's room.

The experiment results are displayed in the Figure 2.4 which represents the differences in the position of the Center Of Pressure (COP) with eyes open a), and eyes closed b) when the subject is on tiptoes with the legs closed, hindering the subject's equilibrium. Both images are displayed with the same axis scale.

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



a) b)

Figure 2.4: Comparison of the COP on the ground in the experimental experiment with eyes open a) and closed b) when a human subject is on ”spout standing” with the legs closed (same scale).

The preliminary results suggest, as expected, that vision has a major role in the postural sway.

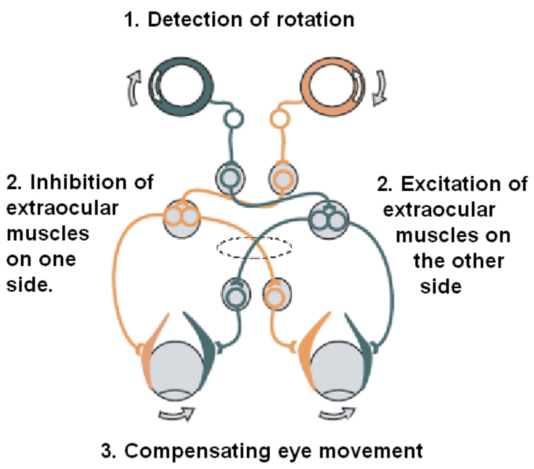
Next sub-chapter will focus on the importance of the vision sense both in humans and some animals in regard to the balance.

2.1.2 Gaze

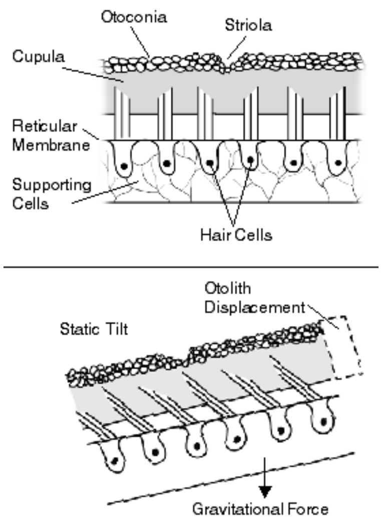
If the scene is slipping across the eyes faster than a few degrees per second, the visual system in the brain is to slow too process that information [62]. Therefore, understanding what is on the human's field of view (FOV) while they move the head, is as much important as the movement of the eyes towards the feature to make an adequate use of the visual ability [26]. The region of the inner ear works with the visual system to keep objects in focus when the head is moving as the Figure 2.5 a) suggests.

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 2.5 b). This is an involuntary process called vestibulo-ocular reflex regulated by the cerebellum [26].

Some animals have better sense of balance than humans. For instance, cats use the tail to play as a counterbalance [61] and most birds eyes, in proportion to there own body size, are much bigger than other animals eyes [63, 4]. This explains why the births 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 the pigeons while they are walking.


1. Detection of rotation
2. Inhibition of extraocular muscles on one side.
2. Excitation of extraocular muscles on the other side
3. Compensating eye movement

a)



b)

Figure 2.5: The vestibulo-ocular reflex. a) The rotation of the head is compensated by the movement of the eyes. b) (adapted from [64]).

Moreover, some birds have the facility to change their cornea's shape, enabling them to have a greater zoom range[21, 64].

Humans are also able to stabilize the head in relation to the enviroment. 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 [65]. Nevertheless, this system is particularly critical in 1–3 Hz range when the head is in resonant instability [40]. 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 proving that impression [8].

As we have seen before, it became evident that vision plays a major role in animals 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.

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

2.2 Vision Based Balance in Robots

In the introduction of this dissertation was mentioned that the balance based on visual-feedback has been introduced in humanoid robots, but the question - how? - remains.

In 2006, Ushida, S. *et al.* emphasized the need of gaze, following the line of humans, arguing that given the fact that the visual sensors are placed in the head of the humanoid robots, it swings violently with the robot's motion, therefore the quality of the acquired images would improve given to reduced disturbances which enables them to better tracking [59].

Oda N. *et al.* in 2011, used a monocular CCD[4] camera and a simple visual target to measure the camera's position deviation which is related to the robot's deviation, as the Figure 2.6 suggests. To get the target position on plane, the CAMShift algorithm, which uses a HSV[5] color feature model, was used to obtain the object position on image [38, 7]. Knowing that $\triangle q_{robt} \propto \triangle v_{image}$, were $\triangle q$ is the variation of the joints and $\triangle v$ the shift between two successive images, is possible to extract the sum of the joint deviations. In 2013, the same author has applied optical flow vector[6] for visual feedback control increasing the stability of ZMP [37].

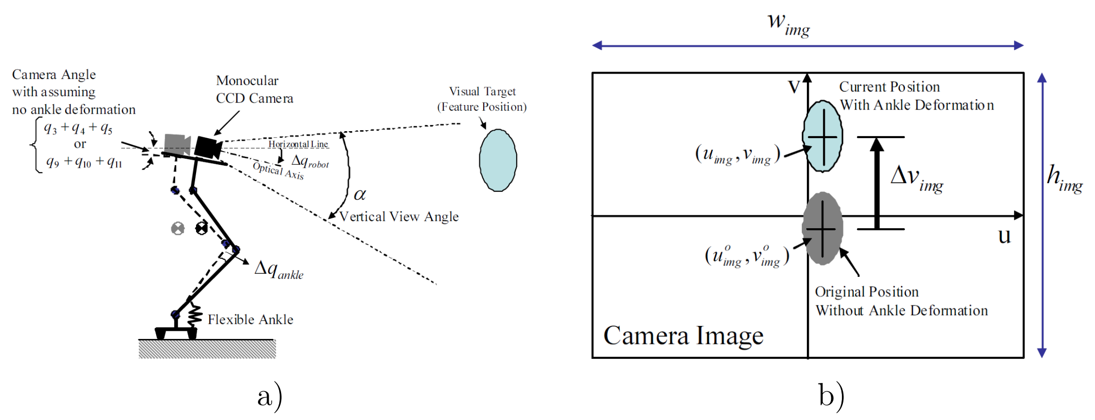


Figure 2.6: a) Visual sensor model (side view) and b) Visual target in camera image (adapted from [38]).

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. In 2009, Chandraker, M. *et al.* brought in a motion estimation using a stereo pair with strain lines as features and infinite lines for indoor environments taking advantage from the edges [10].

[4]Charge-Coupled Device.

[5]Hue, Saturation and Value.

[6]Pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer [64].

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 mapping method to estimate 3D points, that requires a high environment texture[22].

2.2.1 Vision-related works in the PHUA

With regard to the PHUA, Rodrigues, A. ([48]) 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 system 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 the 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 an *framerate* $\sim 25\,Hz$ [48].

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 Pan and Tilt Unit (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[7]. No further study about the performance of the system was reported, specially the tracking system.

2.3 Conclusion

As we could see so far, there is no doubt that vision plays a huge role in humans balance as well as in some animals. In order to perform the PHUA with the visual feedback capability to help its balance, it is mandatory to have gaze. Furthermore, it must have a camera to give reliable and real-time information, good processing capabilities and a responsive mechanical control.

In regard to the PHUA, being an ongoing project, it is mandatory to use the available infrastructure and adapt it to the needs, having always into consideration the future and the other projects that are parallel in the same infrastructure.

The next Chapter describes how the mechanical adaptations were made and how the infrastructure was setup.

[7]It is based on the angle of the servos because it was assumed that the ball is on the floor

Chapter 3

Infrastructure

This Chapter carries a survey about the PHUA's ability to perform and adapt to the introduction of visual-feedback, gathering information that will underpin the infrastructure. Also describes how the needed adaptations to the existent structure were performed, in order to obtain a reliable structure to fulfill the objectives.

3.1 Neck Structure

To ensure the proposed objectives on the first chapter of this Dissertation, the first necessary step was to verify the PHUA's mechanical ability to perform a gaze task, which includes the necessary DOF's to rotate the neck and a vision system.

Given the mechanical structure of the PHUA's regarding those parameters, it was verified that the existent mechanical components were damaged and/or under dimensioned for the propose.

The Figure 3.1 illustrates the state of the art with a CAD[1] design illustration at the beginning of this dissertation where the proposed objectives must be performed.

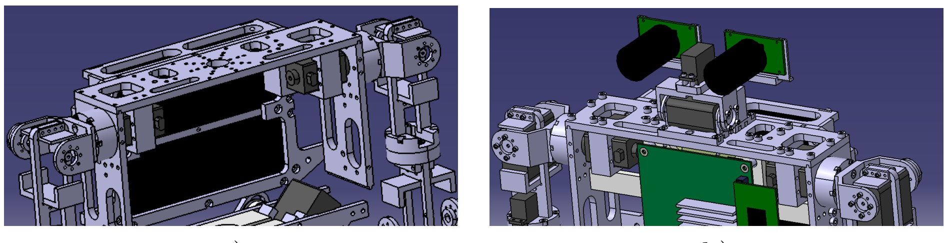


Figure 3.1: PHUA's upper body design before the dissertation starts a) with the neck and b) without the neck (hidden).

[1]Computer-Aided Design.

An assessment of the check-list needs was recorded in order to redesign all the PHAU's neck structure.

To design and build the neck, the following points were considered:

- Mechanical restrictions to the existent PHUA structure, camera and servomotors[2] size;
- Put some parts of the neck inside the truck, reducing the moment induced in the humanoid robot, also augmenting its compactness;
- The manufacturing process when designing the mechanical parts, in order to be feasible to build;
- The FOV of the robot and therefore, the 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 part to select the materials of the components.
- To be compact and robust;

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

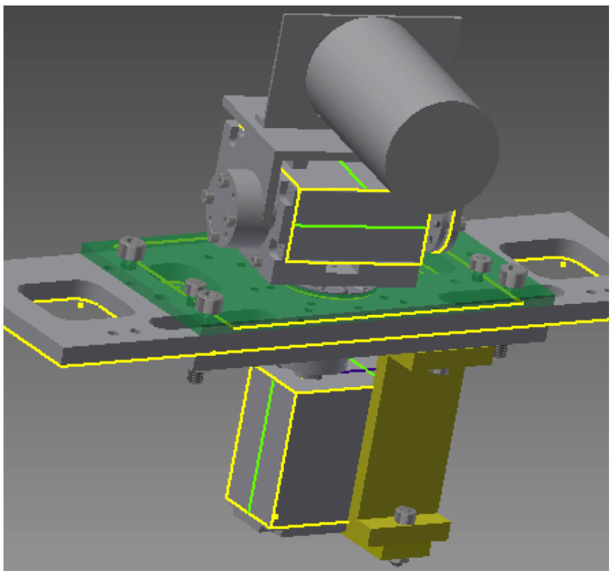


Figure 3.2: PHUA's neck Computer-aided design with software Inventor®.

The *pan-tilt* configuration was maintained comparing with the previous structure, because it combines only two DOF's with the possibility to move in 3 dimensions (X,Y,Z). Regarding the FOV, given that, in fact, this is a humam-like-robot, it was considered the FOV of a human and therefore, it was assumed that the pan movement is 180^o degrees and the tilt movement is 90^o degrees down for the robot can see its

[2]A servomotor is a rotary actuator that allows for precise control of angular position, velocity and acceleration [51].

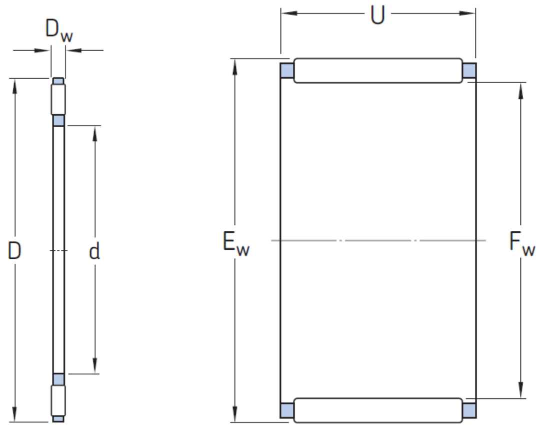
toes, and nearly 30^o degree up, because it is not planed for the robot in future project to look above its head, thus and it is not necessary to increase its dimension (height). Nevertheless, the structure can be easily modified (changing one component) to increase the openness of the tilt movement.

The *pan* servo is built-in on the 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 were considered such as the compactness, it was only played needle rollers requiring that the cage must be included the designed mechanical parts. Thus, the selected bearing is a two axial needle bearing guarantee 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, it could not be used a combined bearing, which is specially designed for that propose, moreover, all the bearings are needle bearings given its compactness. All of those bearings are provided by the company SKF®.

The Figure 3.3 illustrates how the bearings were selected based on the function and the conditions where it has to be applied.



a) b)

Figure 3.3: Types of the bearings applied in the PHUA's neck, a) axial load b) radial load (Adapted from [55]).

To calculate the two required servomotors torque force, 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 ($120\,mg$) are described on the next subsection 3.2.2) and the distance to the center of rotation, the following equations come:

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

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

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

It was considered that both servomotors have the same binary 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 the Figure 3.4 is a photograph after all components were machined and assembled at the DEM factory.

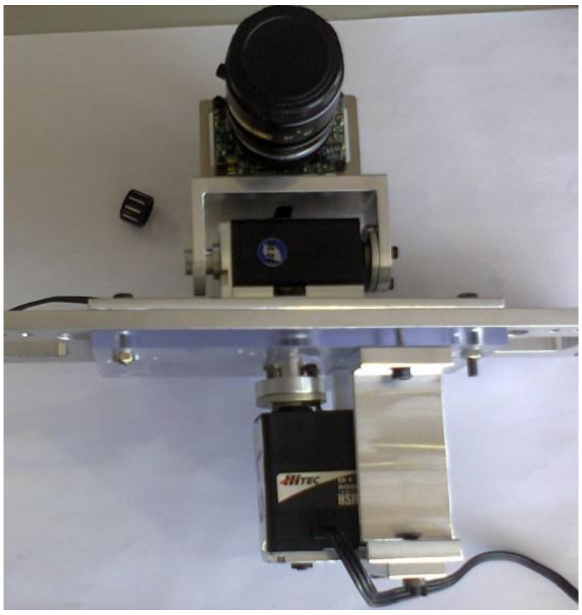


Figure 3.4: Final neck structure after the assembly.

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

3.1.1 PTU's Forward Kinematics

After the structure were made, in order to make a kinematic characterization of the neck structure, it was calculated the neck's forward kinematics following the Denavit–Hartenberg parameters (1955) that may be used to plan or study the movements or even in the future for further improvements and analysis [34].

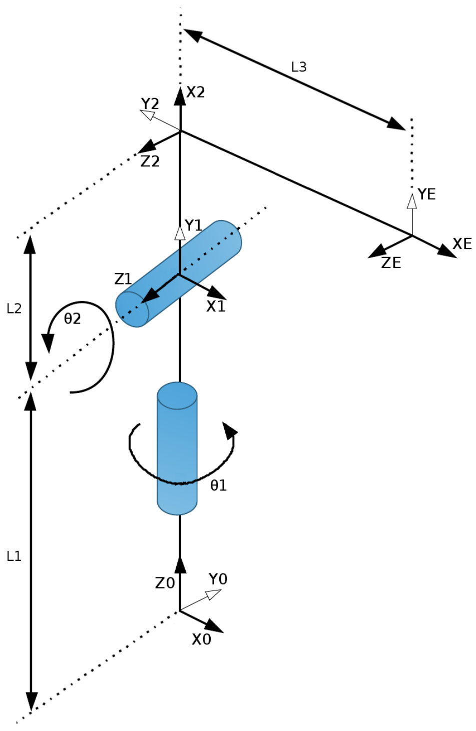


Figure 3.5: Neck's kinematic schematic.

Analyzing the figure 3.5 and following the Denavit-Hartenberg rules, it was obtained the Table 3.1.

Table 3.1: Direct Kinematic 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

Given the Denavit-Hartenberg table, it was applied the equation 3.5 that represents the transformation matrix for each link.

$$A_i = Rot(z, \theta_i) \times Trans(l_i, 0, d_i) \times Rot(x, \alpha_i) \tag{3.4}$$

$$A_i = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & l_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & l_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

Applying the matrix 3.5 for each link T of the neck, the correspondent matrices are given in 3.6.

$$A_1 \begin{bmatrix} c_1 & 0 & s_1 & 0 \\ s_1 & 0 & -c_1 & 0 \\ 0 & 1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A_2 \begin{bmatrix} -s_2 & -c_2 & 0 & -l_2 s_2 \\ c_2 & -s_2 & 0 & l_2 c_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad A_3 \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -l_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.6}$$

Multiplying all matrices, it is obtained the matrix 3.8 that relates the link 0 with the end- effector E, $({}^0T_E)$, which in this case corresponds to the focal point of the camera.

$${}^0T_E = {}^0T_1 \times {}^1T_2 \times {}^2T_E \tag{3.7}$$

$${}^0T_E = A_1A_2A_3 = \begin{bmatrix} \begin{bmatrix} c_1c_2 \\ s_1c_2 \\ s_2 \end{bmatrix} & \begin{bmatrix} -c_1s_2 \\ -s_1s_2 \\ c_2 \end{bmatrix} & \begin{bmatrix} s_1 \\ -c_1 \\ 0 \end{bmatrix} & \begin{bmatrix} l_3c_1c_2 - l_2s_2c_1 \\ l_3s_1c_2 - l_2s_2s_1 \\ l_3s_2 + l_2c_2 + l_1 \end{bmatrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.8}$$

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:

$$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} l_3c_1c_2 - l_2s_2c_1 \\ l_3s_1c_2 - l_2s_2s_1 \\ l_3s_2 + l_2c_2 + l_1 \end{bmatrix} \tag{3.9}$$

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

3.2 Hardware

The main blocks that constitute the infrastructure with regard to the hardware is the camera, the servomotors, the central processing unit (computer), the power supply for the DC servomotors, the FireWire and the $RS-232$ adapters. The following Figure 3.6 represents the main constituent blocks and its interactions.

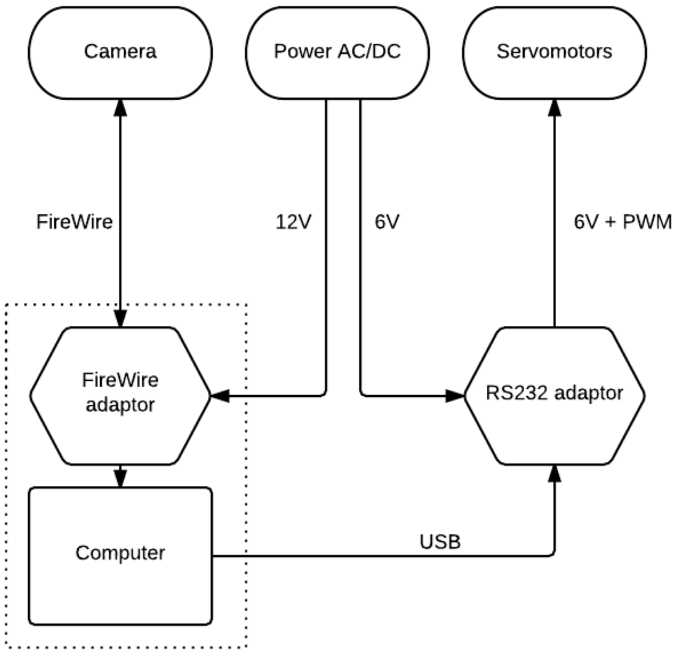


Figure 3.6: Schematic of the hardware connections of the developed system.

The ensuing subsections reports two of the tree main hardware components of the project, the camera and the servomotors.

3.2.1 Servomotors

In line with the whole PHUA witch have digital HITEC® servomotors in all active joints of the humanoid, the ones that were to be 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 force calculated and all the project parameters, including a reduced size/weight, the servomotors selected to control the neck are both $HSR-5498SG$ [28].

The servomotors have a operating voltage range from $6\,V$ to $7\,V$, $0.22\,s/60^o$, $11\,Kg.cm$, steel gears and 180^o of amplitude. This servomotors were made to receive position set-points (SP), nevertheless they allow to change its 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 precision of $0,1^o$. With regard to the velocity value, it is defined in a integer range from 1 to 255 (average velocity) [13]. The servomotor has an internal micro-controller that regulates the PD controller, it can be configured with the software supplied by HITEC® [19, 13].

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) [13].* The Figure 3.7 illustrates the interface board between the computer and the servomotors, as well as the power input (on the bottom of the Figure).

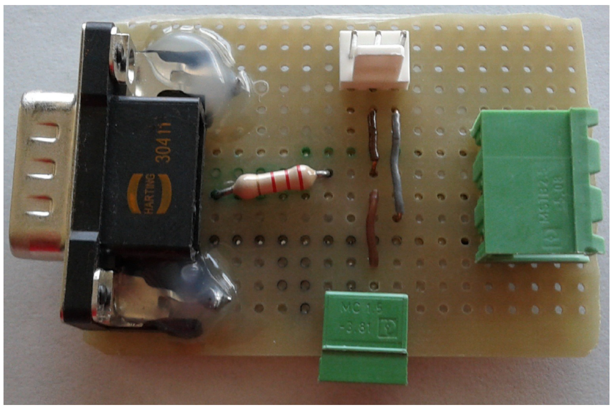


Figure 3.7: Servomotor communication converter board (adapted from [13])

For all the servomotors it was attributed an unique ID that can 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, $120\,mg$ of weight, 60 *fps* and embodies a FireWire 1394*a* communication protocol [35]. 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 the Figure 3.8(Adapted from([3])). The adapter needs an external power source of $5\,V_{DC}$.

The camera's lenses and the camera on its own apply an 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 others,

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

Figure 3.8: Express card FireWire - Unibrain®

such as all the cameras. It is necessary to make an software correction of the distortion in order to get the conditions for the implementation of the computer vision algorithms. One of the big advantages to work on a common used open-source framework with a large online community, the Robot Operating System (ROS[4]), is that much code is available online, with many general purpose packages including the camera calibration package.

Following the instructions of the reference [49] provided by ROS[5], using a chessboard (see Figure 3.9), it was made the image's calibration and obtained the calibration file. The captured image frames subsequently published[6] by the camera capture node are calibrated in this node and are subsequently subscribed by the computer vision nodes that perform the computation base on the images acquired by the camera.

This process of calibration encapsulated in the camera ROS node that publishes the images to the other subscribers properly calibrated. This node gets the calibration parameters from the file generated during the calibration process.

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

3.3 Software

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 standard operating system used in the most recent developments of the PHUA and in whole the

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

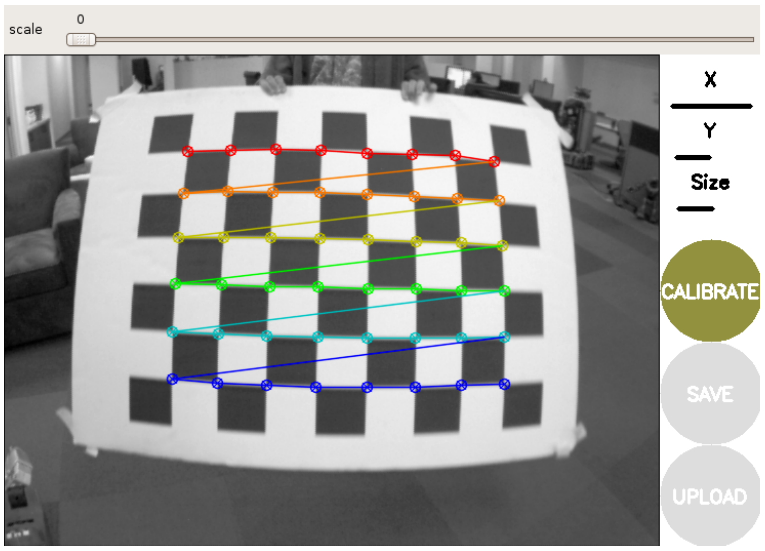


Figure 3.9: *Print screen* of the camera calibration process using ROS *camera_calibration* tool.

LAR.

The LAR has an repository where all the developed software throughout the several projects are available. The software developed in this work will be distributed into two folders. 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 integrates the project must consider the current state of the project, in order to plan their interventions, give modularity to the project, allow to re-utilize code from another sources and integrate new functionalities; in this way, the Robot Operation System (ROS) software has been used in the PHUA since the beginning of the project.

The ROS is a free and open source software framework for robot software development. It is widely used in robotics research having a large on-line support community and it is used in many of the most famous platforms such as NAO and Robonaut 2 [25, 5].

Introducing the ROS to the research community, the authors stated the following on the abstract of the paper [44]:

"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." [44]

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 the 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 [15].

Among many functionalities, ROS is designed to run under Unix/Linux and supports the programing languages *C++* and *Python*, allows peer-to-peer communication between different modules and supply tools for code implementation and testing [19].

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, they may communicate among them selfs with messages [44]. 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 the node *subscriber* [15].

The ROS version employed in this work was the *ROS Hydro* which is the seventh release version[7].

One of the tools that ROS provides is the *rosbag* that enables to store information of the *topics* in *bags* and play them back anytime, allows to process and analyze data off-line. A *rosbag* is an API[8] to manipulate and use the *bags* for reading, writing among other options. A *bag,* is a file format in ROS for storing ROS message data [50]. 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 the topic */camera/image_raw* $\left(3^{rd}\,line\right)$.

[7]http://wiki.ros.org/hydro
[8]Application Programming Interface.

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

Each command must be provided in different command windows; on this dissertation it was used the program *terminator*[9] that allows to display multiple command windows in the same interface.

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. The firmware to acquire the images from the camera as well as the control of the servomotors are available from previous project of the LAR and the online community underpinning the software structure to go straitly to the development 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 [14].

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 cover 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 on the following Chapter 5) [14, 6].

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 *cv_bridge* API being an interface between the ROS *sensor_msgs/Image* message format and the OpenCV, *cv::Mat* data structure. The Figure 3.10 illustrates

[9]https://apps.ubuntu.com/cat/applications/precise/terminator/
[10]http://opencv.org/
[11]Berkeley Software Distribution.

this process[12].

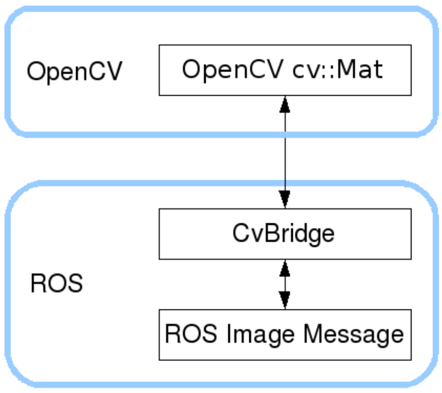


Figure 3.10: Schematic of the interface between the image format of the ROS and the OpenCV, cv_bridge.

The Algorithm 3.1 is the main necessary code to implement the cv_bridge. On this example it is subscribed the topic */camera/image_rect_color* tacking 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 further acceded by the variable *cv_ptr*.

[12]Adapted from http://wiki.ros.org/cv_bridge/Tutorials/

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 it is a software platform that permit to build high-level applications for visual servoing[14]. It was created to bridge the gap that existed in the software environments that allow the fast prototyping of visual servoing tasks, although not dedicated to a certain type of robotic systems, i.e. independent with respect to the hardware [33]. The ViSP is a modular cross platform library implemented in *C++* under Linux, allows to use one or several cameras in order to control the motion of a robotic system (visual servoing [18]) aiming to be extensible, simple and cross-platform [33].

The library of the ViSP platform is divided into three main modules which are the tracking, image processing and further computer vision algorithms, a module for

[13]http://visp.inria.fr/

[14]Specification of a task as a control of visual features [33, 18].

servoing control laws and another for visualization. The Figure 3.11 is a succinct schematic of the architecture of the software library [33].

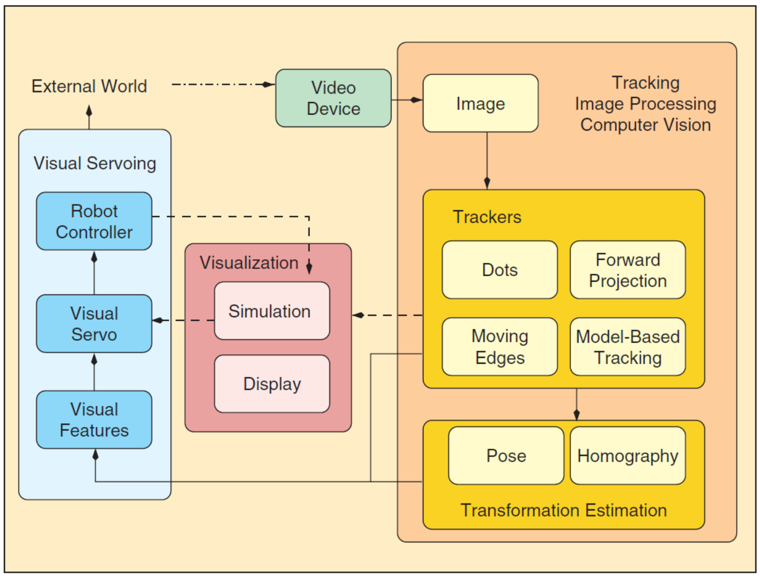


Figure 3.11: ViSP software architecture [33].

In the context of this dissertation, the focus of the implementation of the ViSP is with the intention to provide to the humanoid the capability to track visual targets (fiducial markers).

The ViSP *blob tracking* algorithm, which was the applied one[15], in the first place uses the previous *center of gravity* position of the target from the previous iteration to detect the *boundary* of the *blob* going right, then, follow 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 that the previous iteration, if not, searches in the whole image [42].

The *Freeman Chain* is one of the shape representations[16] that is used to represent a *boundary* by a connected sequence of straight line segments of a specified length and direction. The direction of each segment is coded by using a numbering scheme getting a unified way to analyze the shape of the *boundary*. *Chain Code* follows the contour in counter clockwise manner and keeps track of the directions from one contour pixel to the next [43, 30]. The $C++$ code example 3.2 demonstrates the basic commands to implement a *tracking blob* algorithm using ViSP[17].

[15]The Chapter 4 explains this choice.

[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`

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 makes possible to use ROS in a transparent way benefiting from ROS features such as the Figure 3.12 suggests[18] (similar to the ROS/OpenCV *cv_bridge*). In this way, it is possible to completely separate the code that is specific to the material (frame grabber, robot) or other software tasks, from the one that does the visual servoing.

With the *visp_ros,* it is possible to access the information published on the ROS *topics* such as the image frames publish 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*.

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

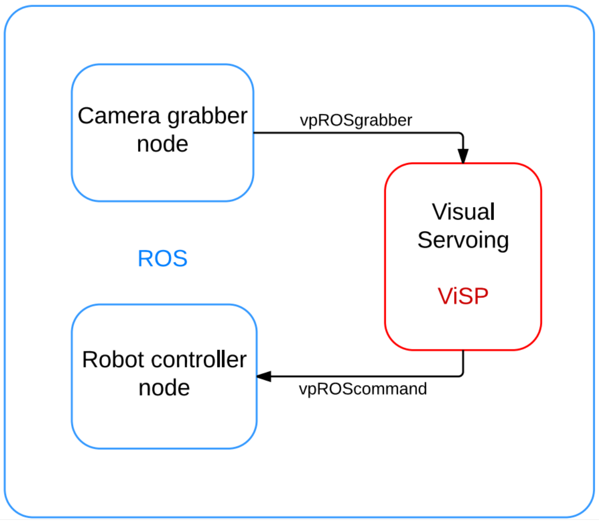


Figure 3.12: *Visp_ros* schematic (adapted from[33]).

Algorithm 3.3 Code example of a *visp_ros* node (adapted from[33]).

```
#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)
     }
}
```

The ViSP is distributed under an Open Source license and it is developed within the INRIA[19] Lagadic project in France[20]. On 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:

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

[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/

Taking advantage of the hardware and software tools presented above, the following section reports the proposed approach to build the desired setup in order to fulfill the objectives of this dissertation.

3.4 Proposed Approach

Given the mechanical setup composed by two servomotors, a camera and the software ROS that allows modular programing, the author's setup proposal follows on the Figure 3.13.

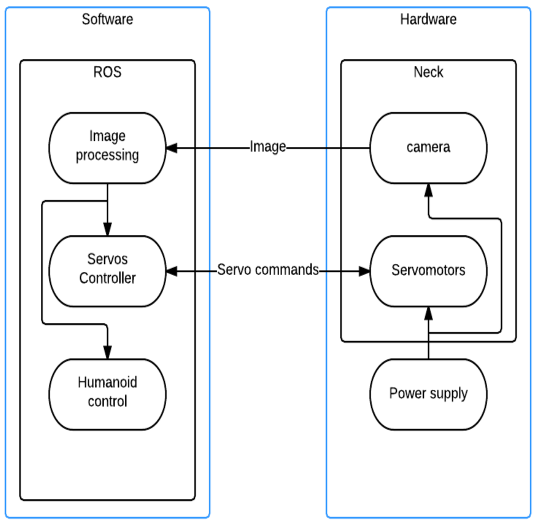


Figure 3.13: Schematic of the proposed approach

As the Figure 3.13 illustrates, the hardware is the mechanical neck composed by the camera, mechanical neck infrastructure including the servomotors and the power supply along with the PC, in accordance to the Figure 3.6.

Based on ROS within several modules, the software is divided in three main nodes, the image processing node module is responsible to acquire an image frame, process the image information and extract the futures to the neck's servomotors controller and the humanoid balance controller. Within the information received by the image processing node, the servos controller node will compute the one information and calculate the instructions to the servomotors based on its control system. The humanoid balance controller will compose visual feedback information with all the other information that it gathers with the other sensors in other to control the humanoid as a unique part.

3.5 Conclusions

As it was refereed before, the PHUA is an evolving project, implying that new modifications must consider the past and the directions of the future towards to an autonomous humanoid in order to guarantee its evolution.

Related to the mechanical structure, it was concluded in the previous sections that a good neck structure is required to allow a good gaze control. Given this, the author proposed to rebuild totally the mechanical structure of the PTU, in order to guarantee a good ground truth and no problems in the future will come from this part. In fact, there is a previous structure on PHUA's neck, but it is not robust, the servomotors have not enough strength and the mechanical structure have some problems in smoothness that is mandatory to not disturb the image acquisition.

In this chapter it was described how the setup was refurbished to accommodate the needs gathering a stable and functional setup to allow the study, application and proof of concept of the control algorithms, such algorithms and conclusions that are described on the following Chapters.

Chapter 4

Image and Control Algorithms

This dissertation proposes that an humanoid robot may benefit from the fixed characteristics existent in the environment around the robot. 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.

Since 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's balance system.

The first section of this Chapter reports the target characteristics that will represent the scenario and the second one refers to the developed vision libraries and how the algorithm was designed to suit the objectives. The third section presents the control algorithms to command the PTU's servomotors and finally it is described the developed MATLAB PTU's tracking simulator and some conclusions.

4.1 Vision

The focus of this dissertation is not to make a wide research on how to extract fixed characteristics in any scenario but rather, one of those is to select a representative one that it will ease the vision feature extraction process and, at the same time, give the possibility to the author 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).

The Figure 4.1 is a schematic summarization of the ROS nodes and its interactions with regard to the computer vision implemented software described in this Chapter.

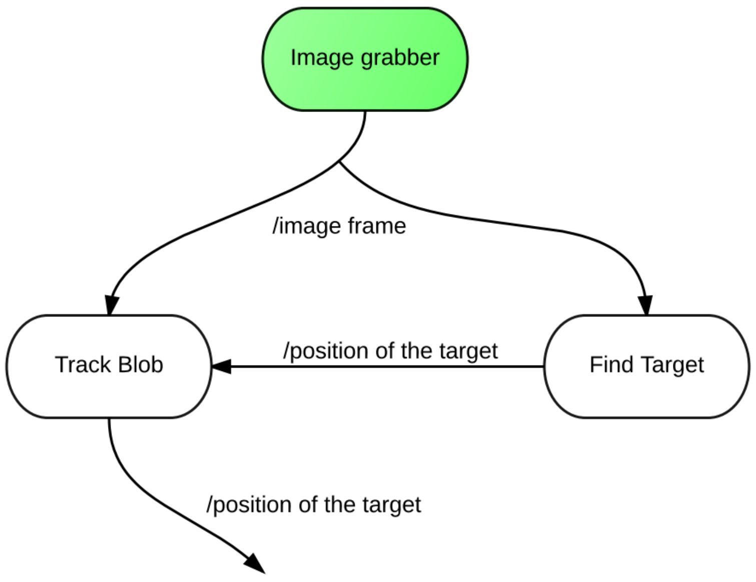


Figure 4.1: Graphical representation of the ROS regarding the image processing.

The *image grabber* is the responsible node 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 distance of the target to the center to the humanoid controller in a topic called */find_arrow_state.*

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 Algorithm B.1 exposes the pseudo-code of the tracking target node.

4.1.1 Target Selection

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.* [38]. Those authors have selected a simple target's shape to

[1]Fixed characteristic on the scene.

be used as a reference for the tracking system (fiducial marker), thus, the selected one is an arrow shape target as the Figure 4.2 illustrates.

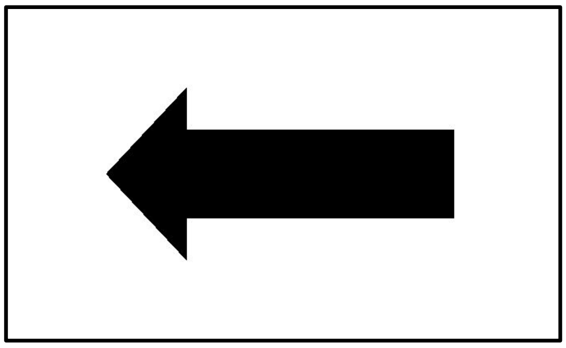


Figure 4.2: Target used as reference for the visual system.

An arrow shape is invariant to rotation and still, as well as fairly easy to extract it 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 maybe confused with other background noises.

The next section explains the algorithm in order to find the target on the PTU's camera image.

4.1.2 Target Detection

To decode the target position on image and 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[2]. 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 an target's shape:

1. The number of line segments (The target (an arrow shape) 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[3];

[2]If it exists on the image!

[3]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[4].

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

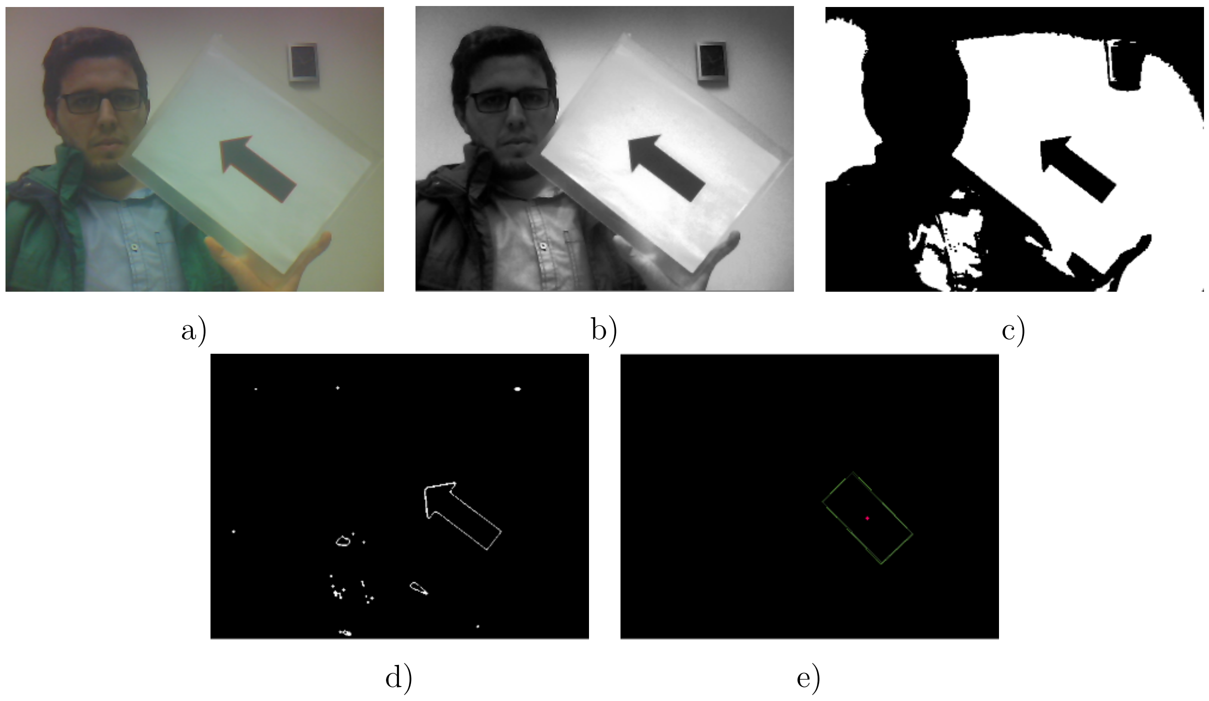


Figure 4.3: Image processing algorithm through the various stages; a) Original color image; b) Grey-scale image; c) Black and White; d) Black blobs contours; e) Target's bonding box.

If the sorting criteria results in one isolated blob, it is assumed that it matches the target and 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.

4.1.3 Blog Tracking

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

Regarding the tracking feature algorithm, ViSP has several algorithms that may be applied depending on the application, such as: blob tracking, key-point tracking,

[4]Most be 5 segments.

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 variant (does not depend on the target's distance form the camera variation). Moreover, ViSP refers that the algorithm is robust if noise and specularity is not too strong [33].

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

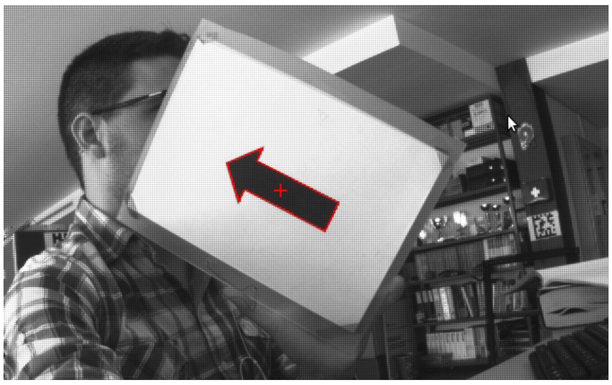


Figure 4.4: ViSP blob tracking.

The implemented algorithm to perform the tracking of the target is based on the ViSP blob tracking algorithm, referred on the previous section 3.3.3 of the 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 (the algorithm throws an exception), i.e., the tracking system can not maintain the target in the FOV irrespectively of the disturbance made by the humanoid; the algorithm re-initializes using the following target position received from the *find target* node. During the process, on each iteration the *track arrow* node sends the position of the target to the PTU controller node.

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

4.2 Control Algorithms

To make the PHUA obtain visual feedback in order to keep its balance, it is necessary to constantly look[5] at a visual fixed target[6] on the scene, so that it can compute the visual information to help extracting its pose and therefore, balance himself. Through the difference between the target's center position and the center of the image together with the implemented mechanical structure described on the previous Chapter, the conditions are gathered to implement a ROS node to control the PTU.

The objective is to keep the PHUA's eyes[7] staring at the target.

On the first approach to compute the visual feedback information to command the PTU's servomotors movements on the control system was assumed that the pan and tilt movements were directly correlated with the position variation of the target in relation to the image center position. Given this simplification, discarding the kinematics of the PTU (although introducing an error), it hugely simplifies the control algorithm[8].

The servomotors have two different parameters to control them, which are the velocity and the position. Therefore, to implement a control algorithm with visual feedback, there are two philosophies/approaches that need to be validated and find the one that best suits the best for the context of this work, which are the control by velocity and the control by position.

The Figure 4.5 is a diagram illustration of 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 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.

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

[5]Keep the camera pointed at.

[6]Keep the target in the middle of the image.

[7]Camera.

[8]On the section 4.3, it is calculated the impact of this simplification.

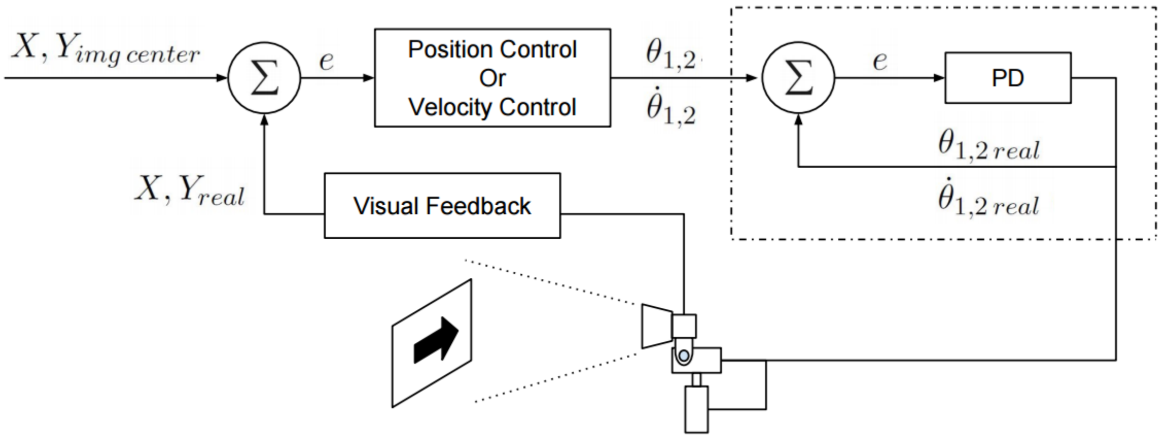


Figure 4.5: PHUA PTU's Control system.

The early approach to be implemented on both control systems is a proportional control (P), which in this context, means that the velocity or the position of the servomotors are proportional to the distance of the SP to the center of the image.

After the development of the algorithm and before the implementation of this approach, it is necessary to divide the controller in two parts. The first part, before any movement of any element such as the humanoid robot or the scenario, the algorithms run with a lower P value and smoothly positions PTU in order to make the camera face the target in the middle of the FOV. After reaching that point (the SP), the second P value is introduced. This is the one that should be tested in order to find the best value to enable the PTU to track the target as fast as possible without entering under instability.

The Algorithm B.2, describes how the PTU controller was designed to face this proposal. This information can be further computed for the PHUA's balance system representing the intended visual feedback (gives the direction of the target to the PHUA's system in relation to the world reference).

4.3 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 matrix laboratory - MATLAB - 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 (taking advantage of the equations from the section 3.1.1) 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, generating 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.

The following sub-section explains how the equations to build the simulator were calculated.

4.3.1 Inverse Kinematics

As mentioned before in section 3.1.1, 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 do 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 4.3 shows the PTU's forward kinematics equation of the ${}^{0}T_E$ obtained in the previous Chapter 3.

$$
{}^{0}T_E = \begin{bmatrix} \begin{bmatrix} c_1c_2 \\ s_1c_2 \\ s_2 \end{bmatrix} & \begin{bmatrix} -c_1s_2 \\ -s_1s_2 \\ c_2 \end{bmatrix} & \begin{bmatrix} s_1 \\ -c_1 \\ 0 \end{bmatrix} & \begin{bmatrix} l_3c_1c_2 - l_2s_2c_1 \\ l_3s_1c_2 - l_2s_2s_1 \\ l_3s_2 + l_2c_2 + l_1 \end{bmatrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}
$$

Throughout the previous equation 4.3, interpreting the proposed PTU's structure illustrated on the Figure 3.5, the position of the end-effector of the PTU[9] on the *ZZ*

[9]Camera's Focal Point.

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 the Figure 4.6 and the previous forward kinematics equation 4.3, the θ_1 is calculated by the equation 4.4.

$$\theta_1 = \arctan\left(\frac{PX}{PY}\right) = \arctan\left(\frac{l_3c_1c_2 - l_2s_2c_1}{l_3s_1c_2 - l_2s_2s_1}\right) \tag{4.4}$$

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

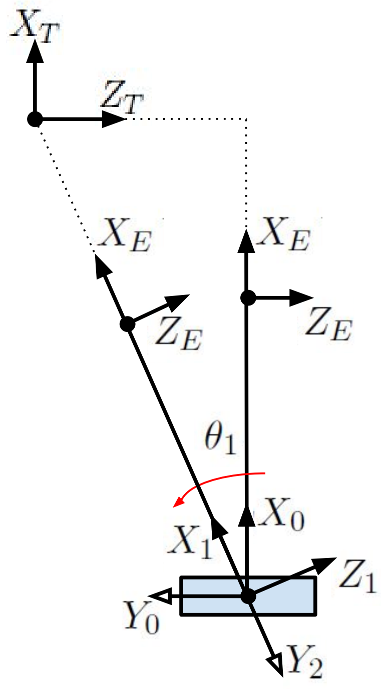


Figure 4.6: Schematic used to calculate the PTU's θ_1 (view from the top).

After the calculus of joint θ_1, the PTU and the target can be represented on a plane such as the Figure 4.7 portrays.

By analyzing the Figure 4.7, 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 4.5.

$$\overrightarrow{r} = \overrightarrow{a} + t \cdot \overrightarrow{b} \tag{4.5}$$

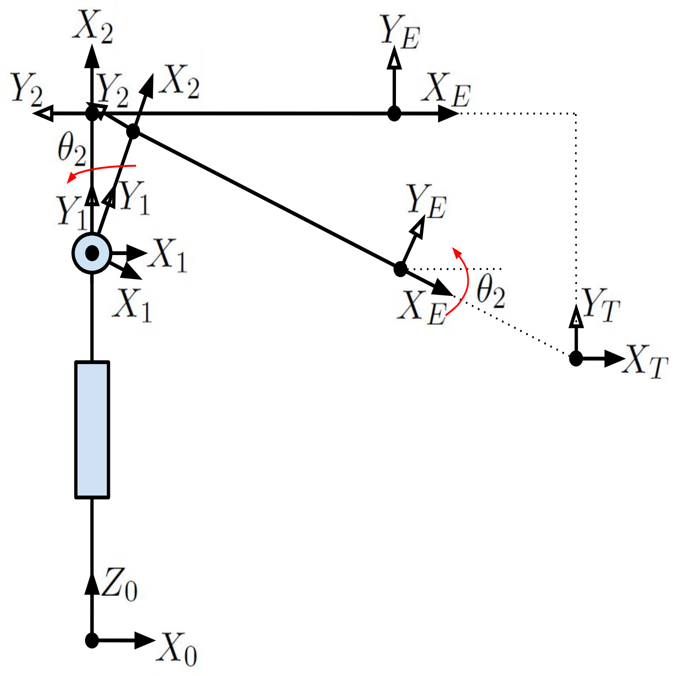


Figure 4.7: Schematic of the plane that contains the PTU and the target.

Applying the equation 4.5 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 4.3 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 4.5, it is obtained the equation that relates any point of the vector $\overrightarrow{r}$ with PTU's reference (0):

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} -l_2s_2c_1 \\ -l_2s_2s_1 \\ l_2c_2 + l_1 \end{bmatrix} + t \cdot \begin{bmatrix} c_1c_2 \\ s_1c_2 \\ s_2 \end{bmatrix} \tag{4.6}$$

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 4.7.

$$\begin{bmatrix} X_T \\ Y_T \\ Z_T \end{bmatrix} = \begin{bmatrix} -l_2 s_2 c_1 \\ -l_2 s_2 s_1 \\ l_2 c_2 + l_1 \end{bmatrix} + t \cdot \begin{bmatrix} c_1 c_2 \\ s_1 c_2 \\ s_2 \end{bmatrix} \tag{4.7}$$

$$=> t = \frac{X_T + l_2 s_2 c_1}{c_1 c_2} = \frac{Y_T + l_2 s_2 s_1}{s_1 c_2} = \frac{Z_T - l_2 c_2 - l_1}{s_2} \tag{4.8}$$

The solution for the previous equation 4.8 can be achieved throughout the manipulation of itself, to obtain an equation form such as the 4.9. This equation has a known solution that is demonstrated on 4.10.

$$K_1 \sin(\theta) + K_2 \cos(\theta) = K_3 \tag{4.9}$$

$$\theta = \arctan\left(\frac{K_1}{K_2}\right) \pm \arctan\left(\frac{\sqrt{K_1^2 + K_2^2 + K_3^2}}{K_3}\right) \tag{4.10}$$

Therefore, manipulating the equation 4.8 in order to have the form of 4.9, 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 whose 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 4.10 consonant with the signal of X_T and Y_T respectively.

Taking advantage of the computation of the forward and inverse kinematics, the Figure 4.8 illustrates a screenshot on a given moment of a linear movement in one simulation example: on a) it is illustrated the trajectory in a red doted line, the blue

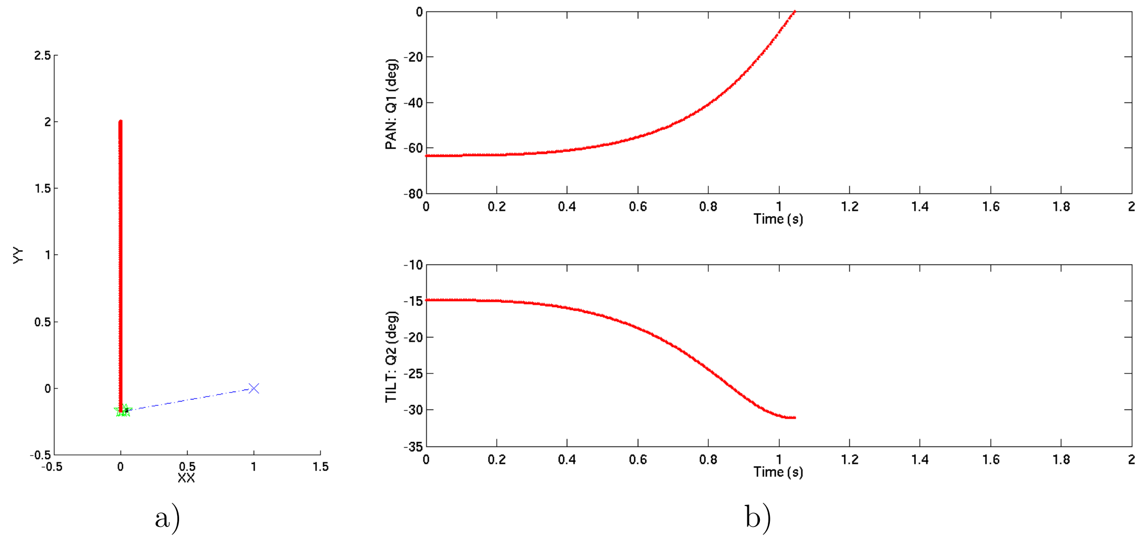


Figure 4.8: PTU movement MATLAB simulator. a) Illustration of a PTU's linear movement on the Y axis in relation with the global reference. b) Pan and tilt angles over time during the movement a).

'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.

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.

4.4 Conclusions

At this point, the humanoid can search the target on its FOV such as to the human eyes, and track it aided by its PTU controller, similarly to 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 balance 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 became available a *roslaunch* for the camera that can be used latter in different contexts.

The 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 propose to validate the reformulated mechanical structure of the PHUA's neck, the image processing and the tracking algorithms. The first section of this Chapter discloses how the experimental setup was structured to develop the experiments, the second section reports how the tests were conducted and the experimental results; the third section analyses the results, making use of the developed simulator and the final section addresses a discussion about the whole work of this dissertation.

5.1 Experimental Setup

The objective of this section is to describe the setup to conduct the experiments in order to test and classify the performance of the developed tracking system.

The simulation of the PTU's 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 this experiments was an anthropomorphic six degrees of freedom robot - Fanuc M-6iB6s - represented on the Figure 5.1, capable of manipulating up to 6kg of payload.

Since the robot Fanuc is calibrated and has a repeatability of 0,08mm [12], 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 different types of movements such as linear movements, the target in a considerable period of time (produce recorded trajectories.

A C++ client application was developed to acquire the data from the robot and

Figure 5.1: Fanuc M-6iB6s used for the experiments [11].

control its movements through a TCP/IP[1] connection. With a 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[2] 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 desired 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.

With Regard to the possessor 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.

The Figure 5.2 illustrates the ROS nodes and the message interactions during the tests. To trigger the experiment synchronously, it was used the PC's keyboard that triggers an ROS message, initializing the FANUC movement, as well as the recording of the data from its position, the computer vision messages data and the instructions given

[1]Transmission Control Protocol/Internet Protocol.
[2]Comma-separated values.

to the servomotors, gathering information to further study and analyze the experiment. The time of computer controller was used as an reference to synchronize all the recorded data.

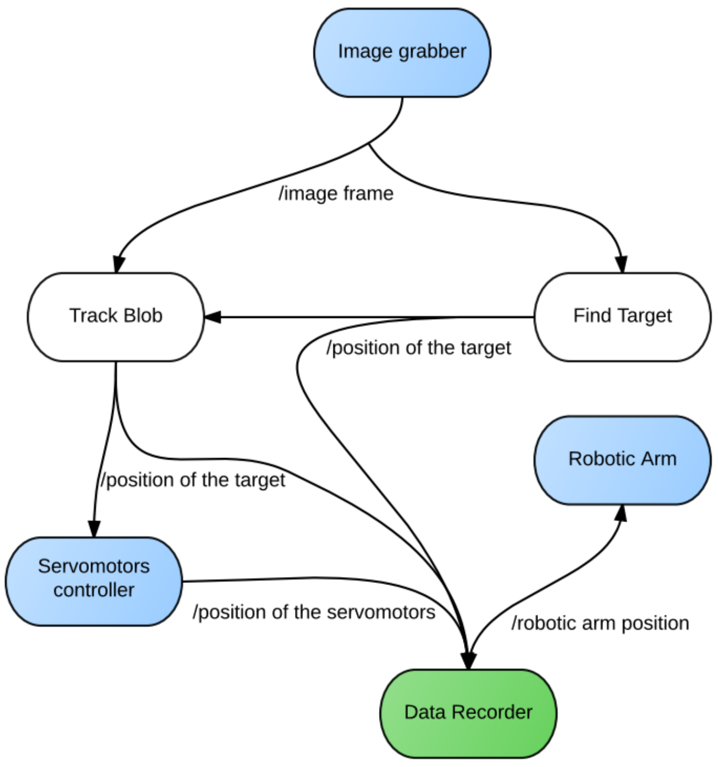


Figure 5.2: Schematic of the software architecture for the experimental setup to test the developed system.

All the recorded data was saved or converted into .csv files for further analysis with a MATLAB script.

5.2 Experimental Results

Based on the previous described experimental setup on section 5.1, this section reports the experiments and its 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 the visual tracking algorithms. The experiment was carried out attaching the target on the Fanuc's end-effector and laying the camera in a such 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.

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 the FANUC arm, as the Figure 5.3 represents.F

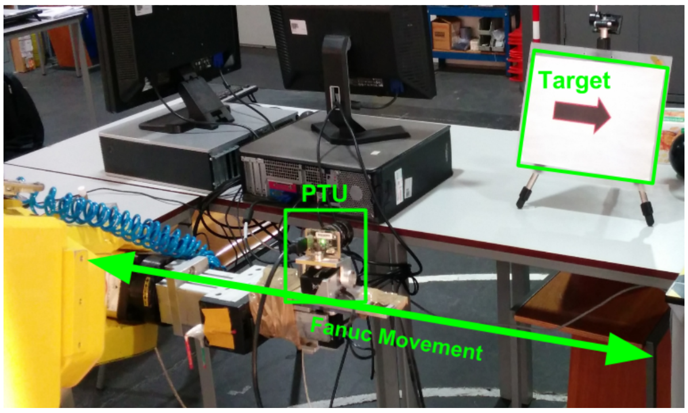


Figure 5.3: Schematic of the experimental setup.

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 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.

The next subsection 5.2.1 addresses the employed metrics to compare and characterize the results of the experiments. The subsections 5.2.2 and 5.2.3 reports the first and second tests respectively. It is presented how the experiments where performed, the movements or theFormula y of each experiment.

5.2.1 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 the this context they are considered errors) since the differences are squared. The equation 5.1 was applied to calculate the MSE. The $\triangle t$ suits for the time period of the experiment, 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.

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

An additional method that was applied to measure the performance with regard to the tracking control evaluation, was the time of stabilization, which suits for time required for the tracking controller stabilize right after the movement of the Fanuc robot stops; the lower the time, the better.

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

The following subsections 5.2.2 and 5.2.3 present the experimental results for the image processing algorithms and for the developed tracking system, respectively.

5.2.2 Image Processing 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 with the actions of the head, producing a displacement of the target on the image.

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 ensued 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 and the other two where 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[3]; the right side relates to the results of the computer vision algorithms. The values of the both left and right yy axis are related to the initial position of the experiment.

The Figure 5.4 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

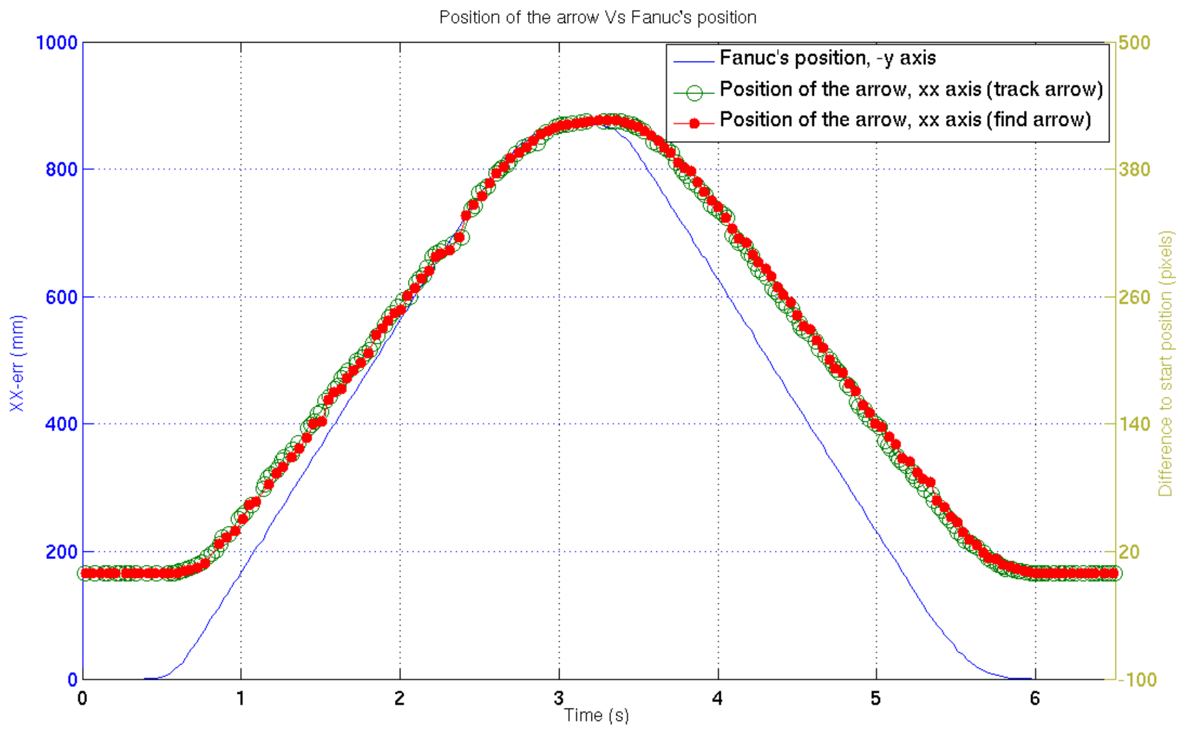
[3]The axis with the most relevant information (along which the movement is more significant).

find arrow node data, regarding the target's *yy* position on the image, presents some variations in the position of the center of the target. Although the variations are not reflected on the *track arrow* algorithm, they depend on the data provided by the *find arrow node*.

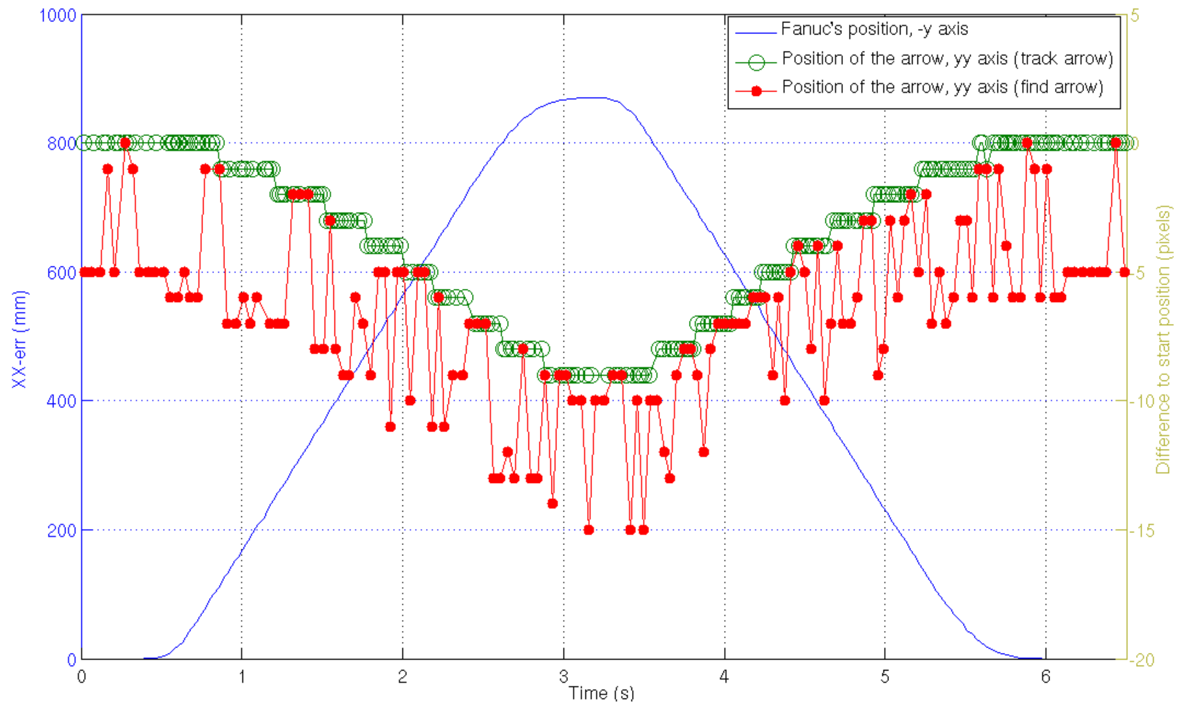
The Figure 5.5 represents the data for the experiment with the maximum velocity of the Fanuc robot $1,120\,m/s$. The behavior is similar to the previous experiment and once again it is noticed that the *track arrow* is smoother than the *find arrow*.

Illustrating the experiment with the maximum velocity of the Fanuc robot $1,814\,m/s$, the Figure 5.6 shows that at this velocity, the algorithm "loses" the target in a considerable period of time $(7,5s\,to\,7,8s)$. 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 $8s\,to\,8,6s$ the *find arrow* loses 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 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.

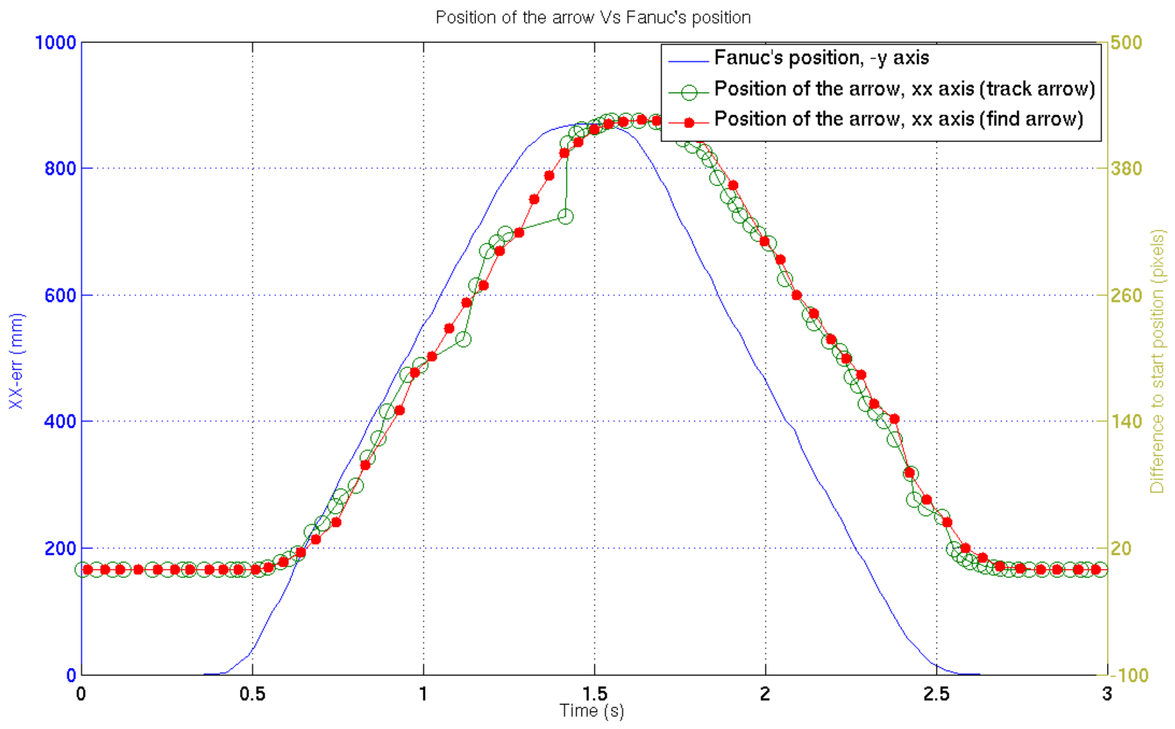


a)

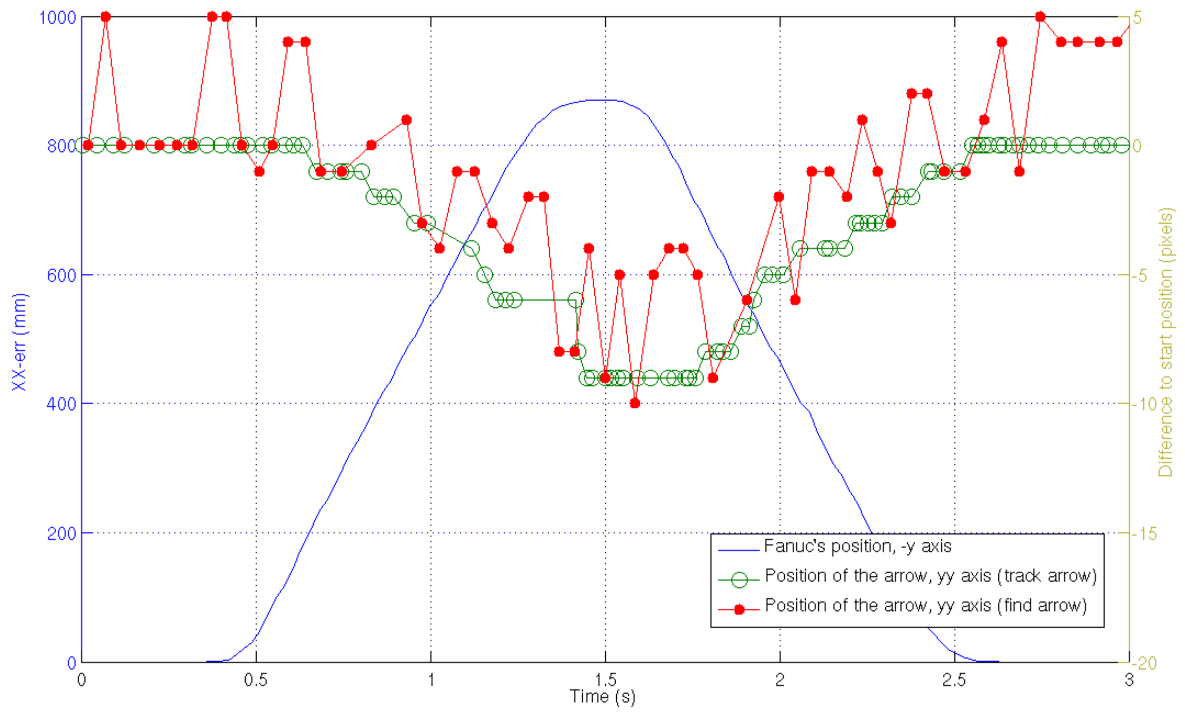


b)

Figure 5.4: 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 target center position on the image.

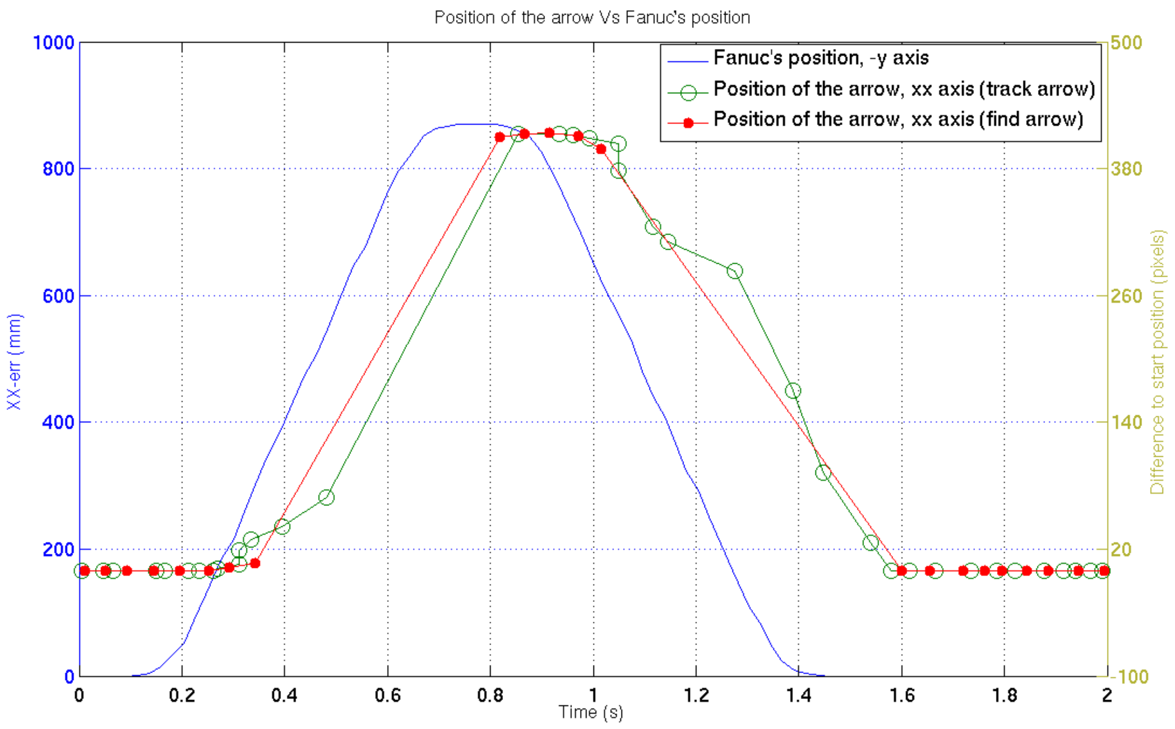


a)

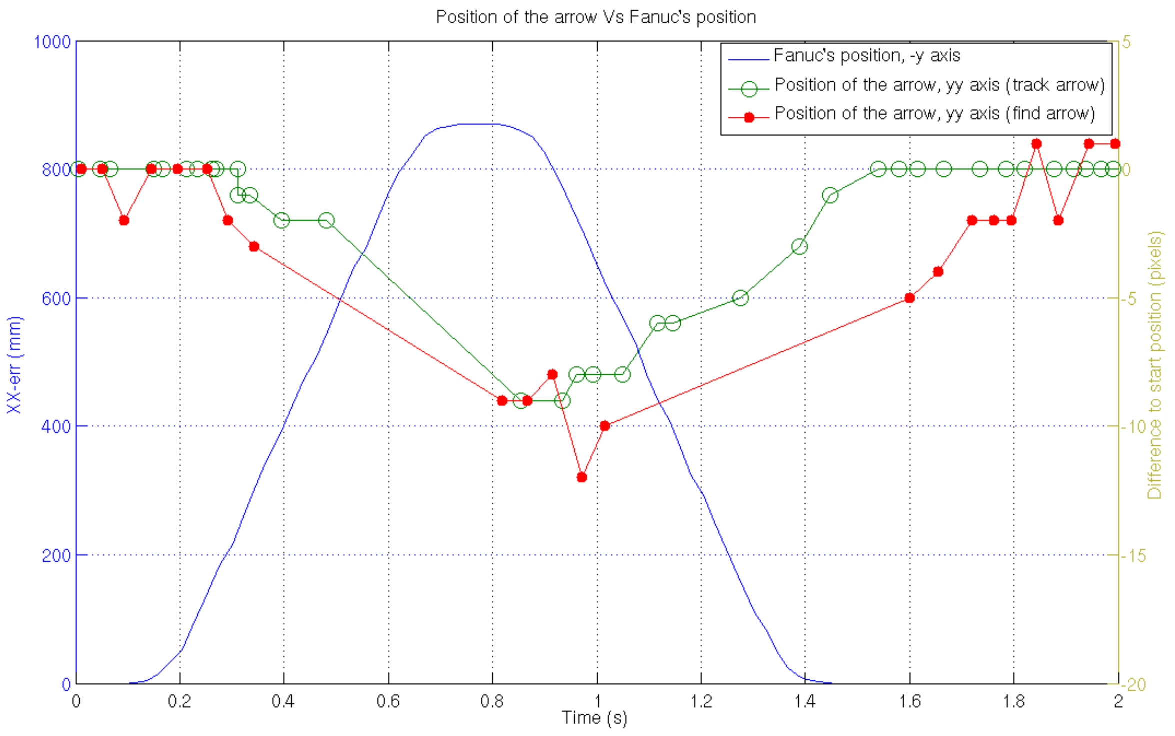


b)

Figure 5.5: 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 5.6: 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.

The Figure 5.7 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 $0,03s$ $(33\,Hz)$.

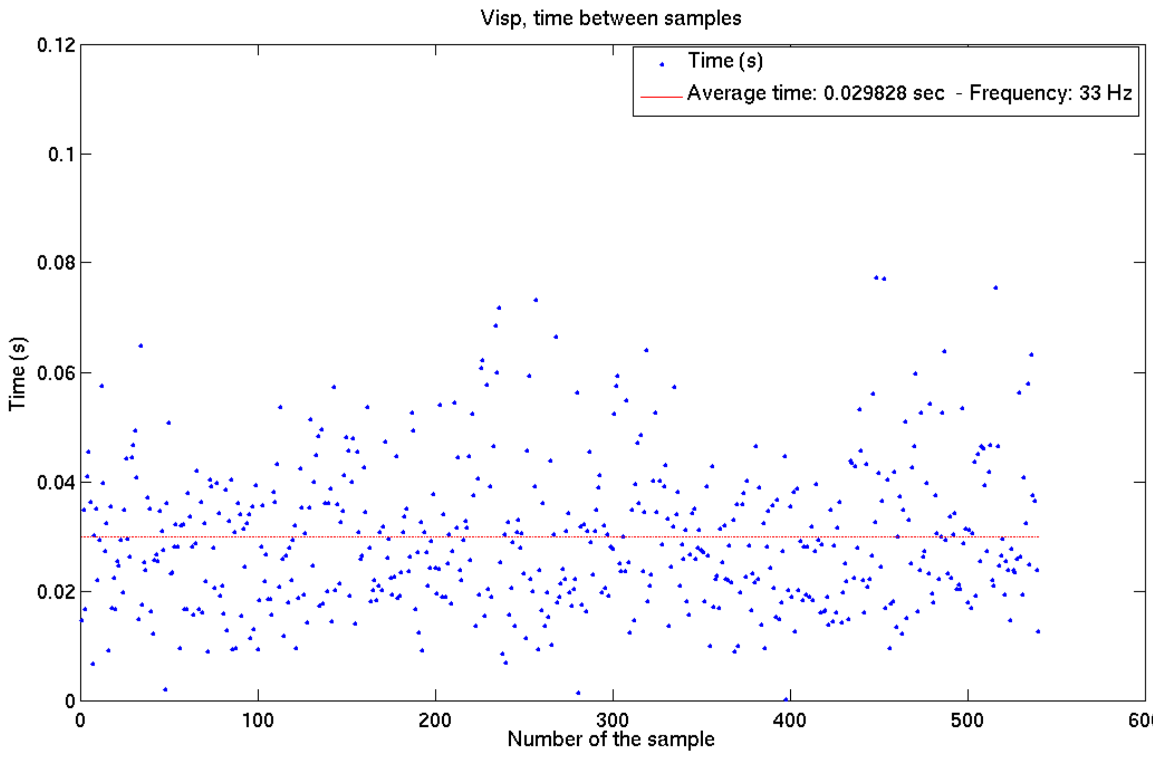


Figure 5.7: Example of the time between target's position values of the "find arrow" ROS node.

The Table 5.1 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 then the *find arrow* node. Moreover, it is verifiable on the illustrated experiments that the frequencies does not present relevant variation between the experiments.

Table 5.1: 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 then the *find arrow node*, which proves the advantage to use a tracking algorithm.

Related to the images from the camera (frames), as expected, when the only active node is the image grabber, it is possible to acquire images at the max frame rate, 60*fps,* nevertheless, along with the activation of the several nodes, the frame rate has decreased substantially.

Taking advantage of the results described in this subsection, the next one reports the experiments made to evaluate all the PTU's tracking system.

5.2.3 Tracking Control Evaluation

Based on the experimental setup described on section 5.1 and the computer vision limits tested on the previous subsection, this subsection describes the experiments and 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 perform the tracking of 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 the 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 the Figure 5.11 (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 the direction;
- Movement B, illustrated for instance on the Figure 5.9 (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 the Figure 5.8 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[4]).

The objective of this movements are to simulate the potential motions of the 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 $72s$ to $87s$ in the Figure 5.8, 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

[4]Corners

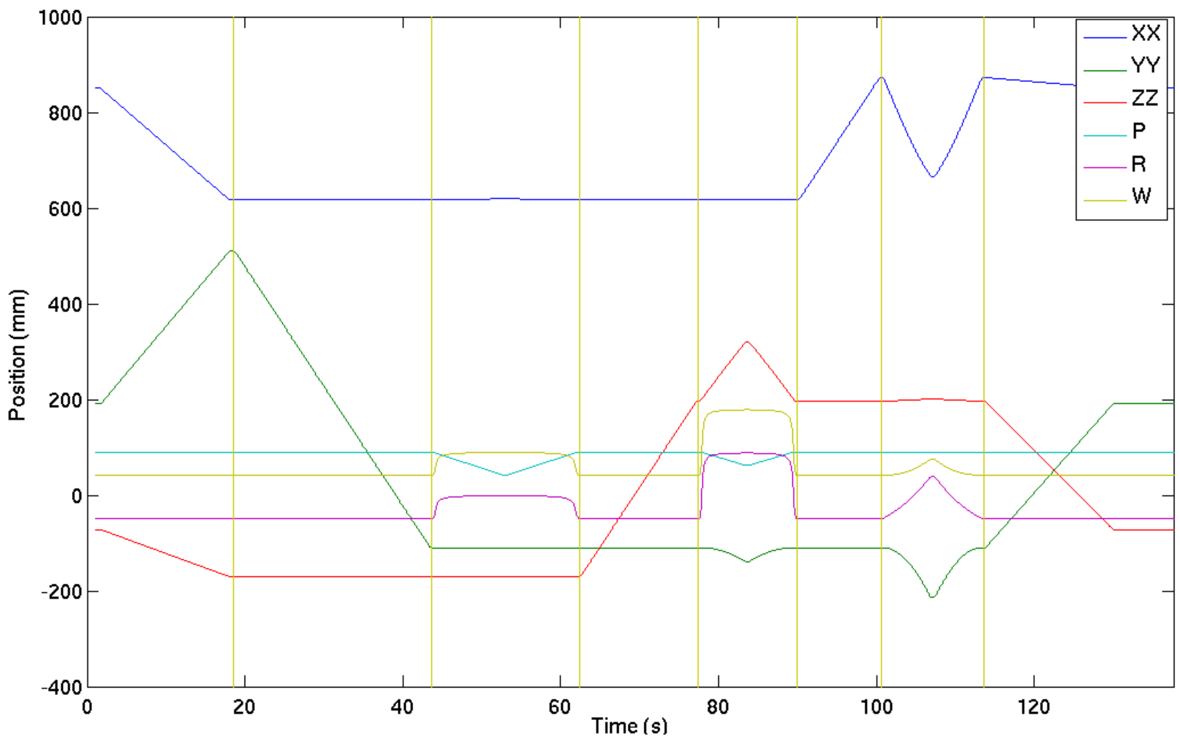


Figure 5.8: Movement C profile of the Fanuc robot end-effector over time.

this movement have vertical yellow lines separating each segment of the movement in order to assist the visualization (see Figure 5.8).

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

The Table 5.2 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 5.2: 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 the MSE, 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 5.2, the experiment 2) suggest that when the K_p is smaller, besides the 'Fanuc velocity' is bigger, the 'stabilization time' is smaller than the 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.

The Table 5.3 presents the data obtained from the experiments with the position tracking control.

Table 5.3: 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)	$2,5$	$2,1$	$6,3$	$1,1$

The Table 5.3 shows that besides the MSE values are higher then the velocity tracking control (see Table 5.2) the 'stabilization time' is lower for the experiences 5) and 6).

The Table 5.4 presents the frequency of the published data during the experiments.

Table 5.4: Frequency results of the Data acquisition on the control by Velocity 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	$38,1\,Hz$	$37,8\,Hz$	$37,9\,Hz$	$37,0\,Hz$	$37,9\,Hz$	$38,3\,Hz$	$38,3\,Hz$	$37,1\,Hz$

The previous Table 5.4 demonstrates that there was not 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 the subsection 5.2.2; nevertheless, the performance of the node 'find target' is constantly inferior to the 'tracking' one.

The Tables 5.5 and 5.6 are a resume of the data indicated above.

Table 5.5: Data comparison of the MSE values for the 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 then the velocity control in for the data presented on the Table above 5.5 and 44% higher on the Table 5.6.

Table 5.6: 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 %

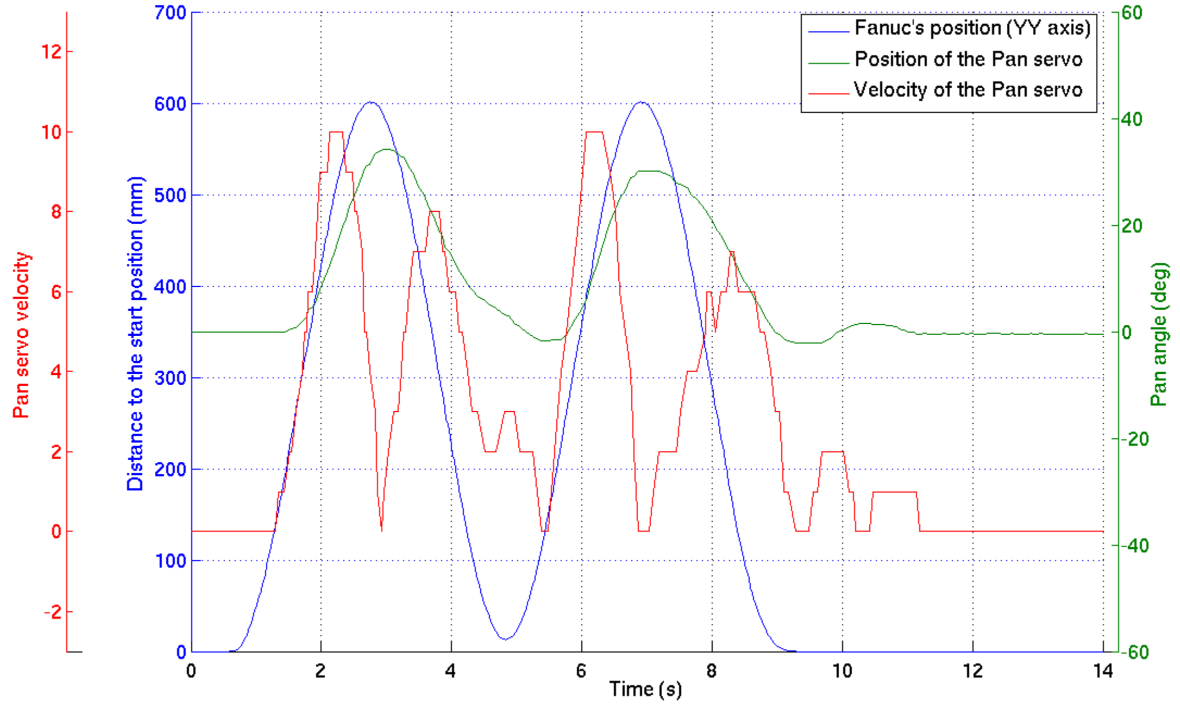
With respect to 'stabilization time', besides the difference is not pronounced, the Table 5.7 shows that the time is superior in the velocity control.

Table 5.7: Comparison of the sum of the 'stabilization time' data at the end of each experiment.

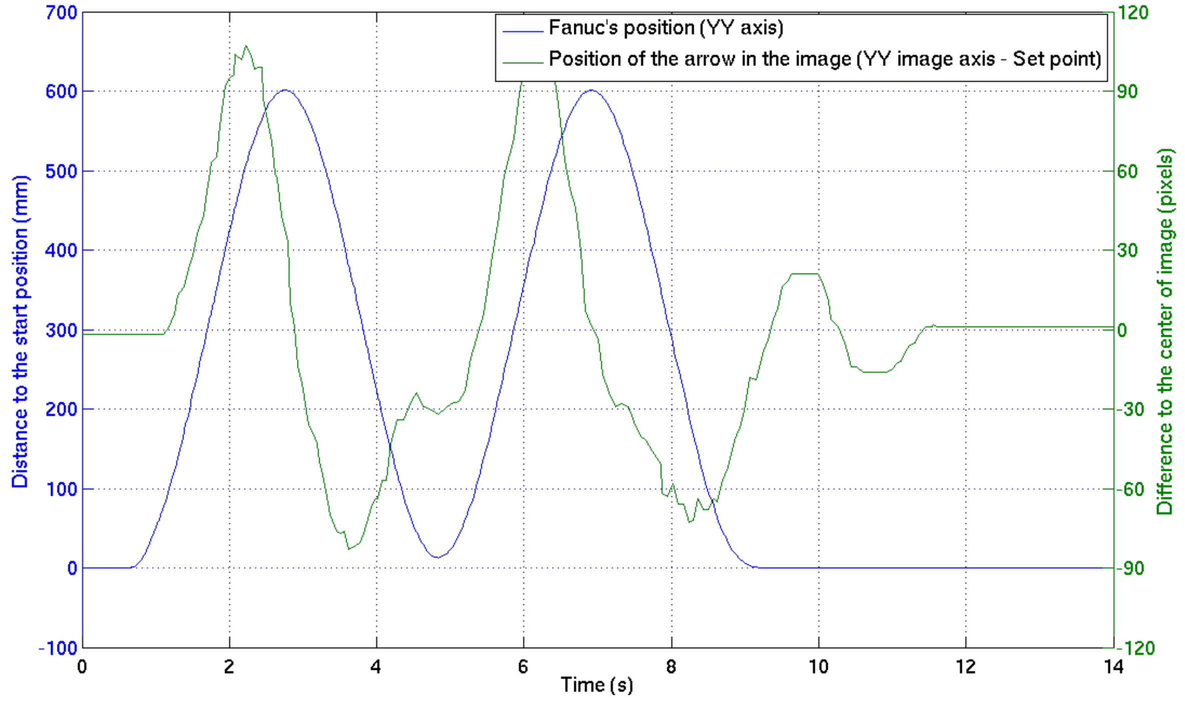
	Velocity Control	Position Control
Time sum	$15,53$	$12,02$
Percentage	56 %	44 %

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

The Figure 5.9 and the Figure 5.10 is 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.

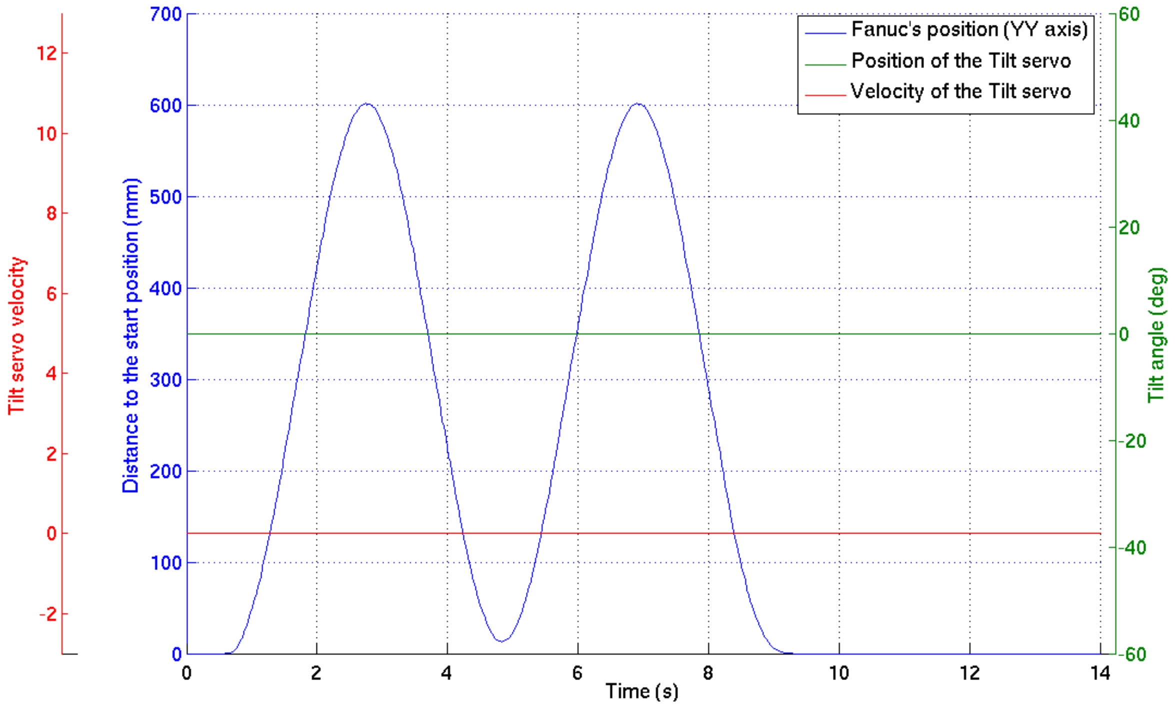


a)

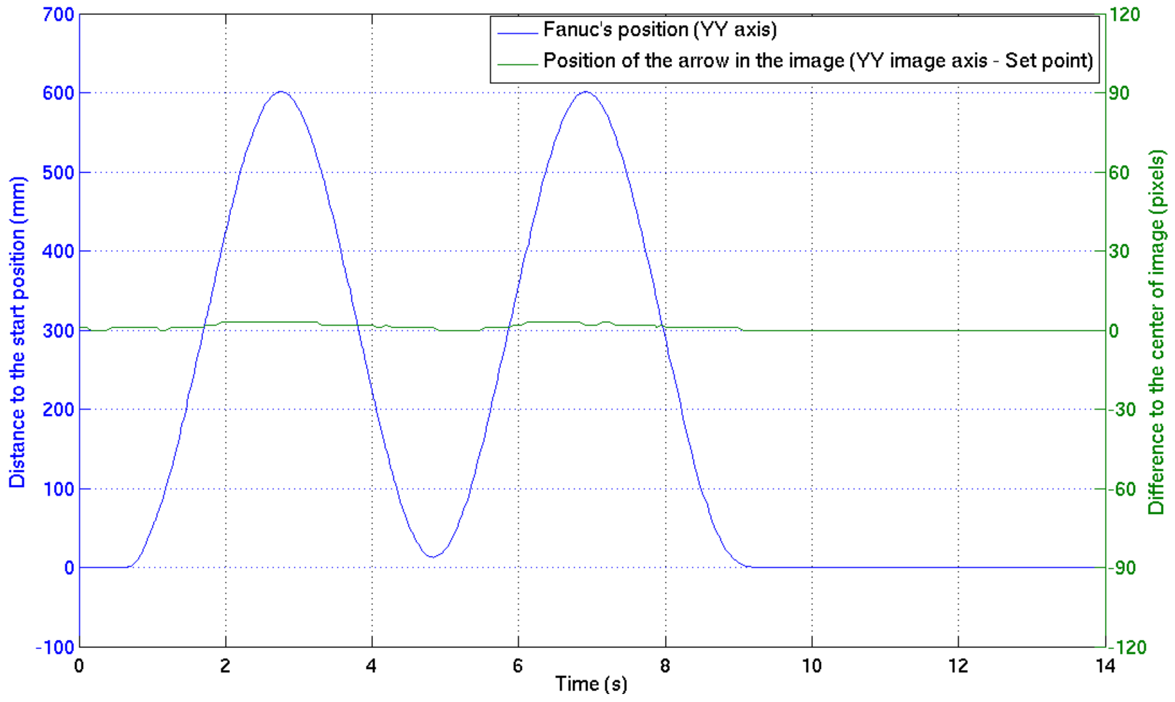


b)

Figure 5.9: 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)

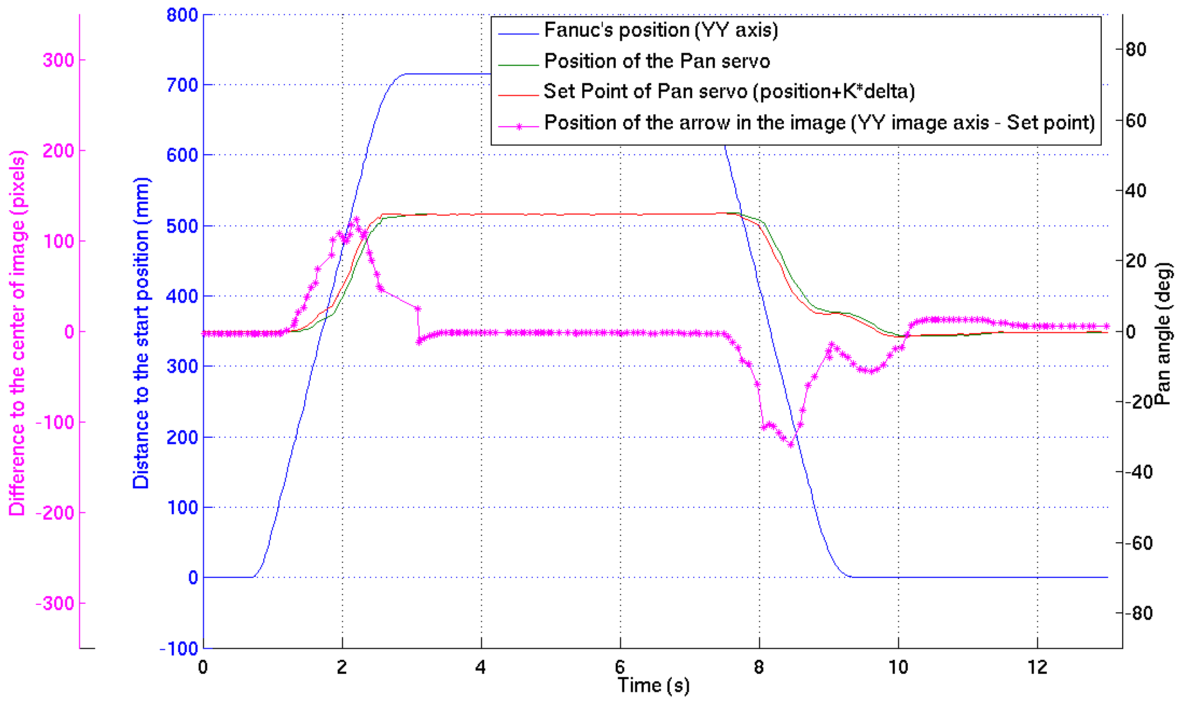


b)

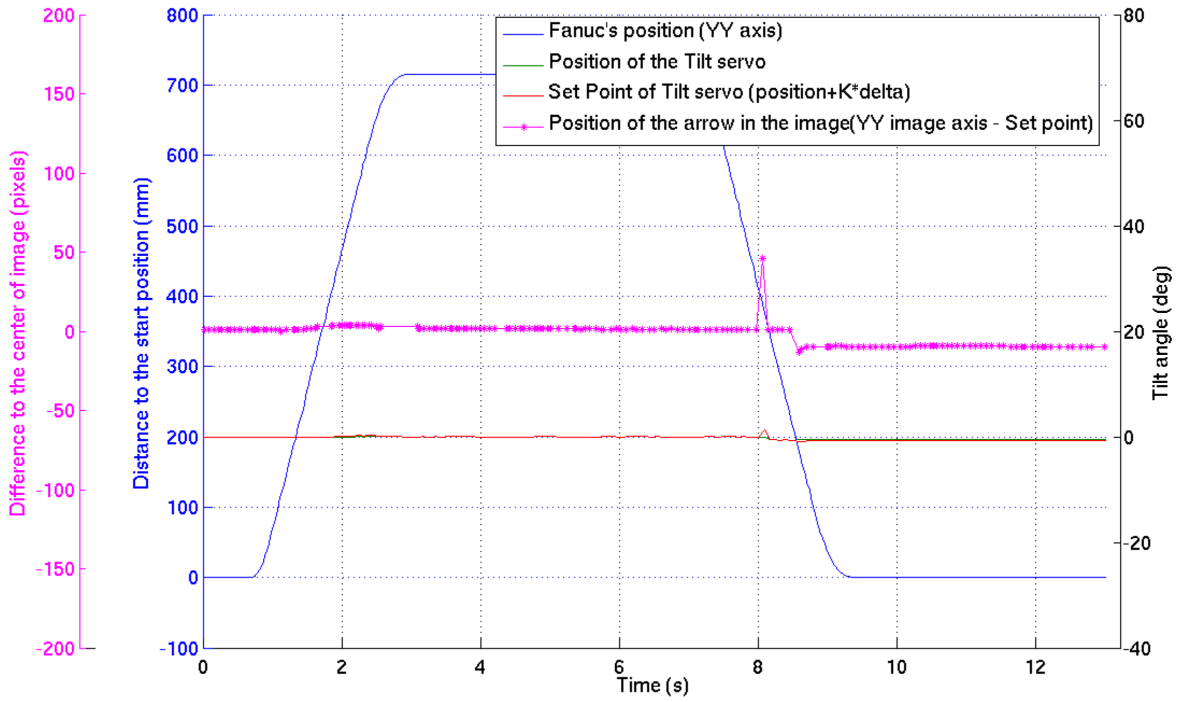
Figure 5.10: Graphical output of the tilt data acquired during the experiment 5), which is a position control tracking with the movement B. The graphic a) relates to the data of the PTU's servos position and velocity; b) illustrates the distance of the target in relation with the center of the image over time. Both illustrations present the Fanuc robot movement profile on the *yy* axis as a comparison "pattern".

The Figure 5.9clearly illustrates the 'stabilization time' and allow to analyze the performance of the PTU tracking system over time $(20, 2s\, to\, 22, 45)$. Analyzing the data of the position of the PTU's servomotors verses 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. The Figure 5.10 does not report any relevant data; since the movement on the tilt component is negligible, i.e, the curves remain roughly flat.

The Figure **??** presents a graphical representation of the 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 5.11: 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, the Figure 5.12 shows the distribution of the 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 the previous Figure5.11).

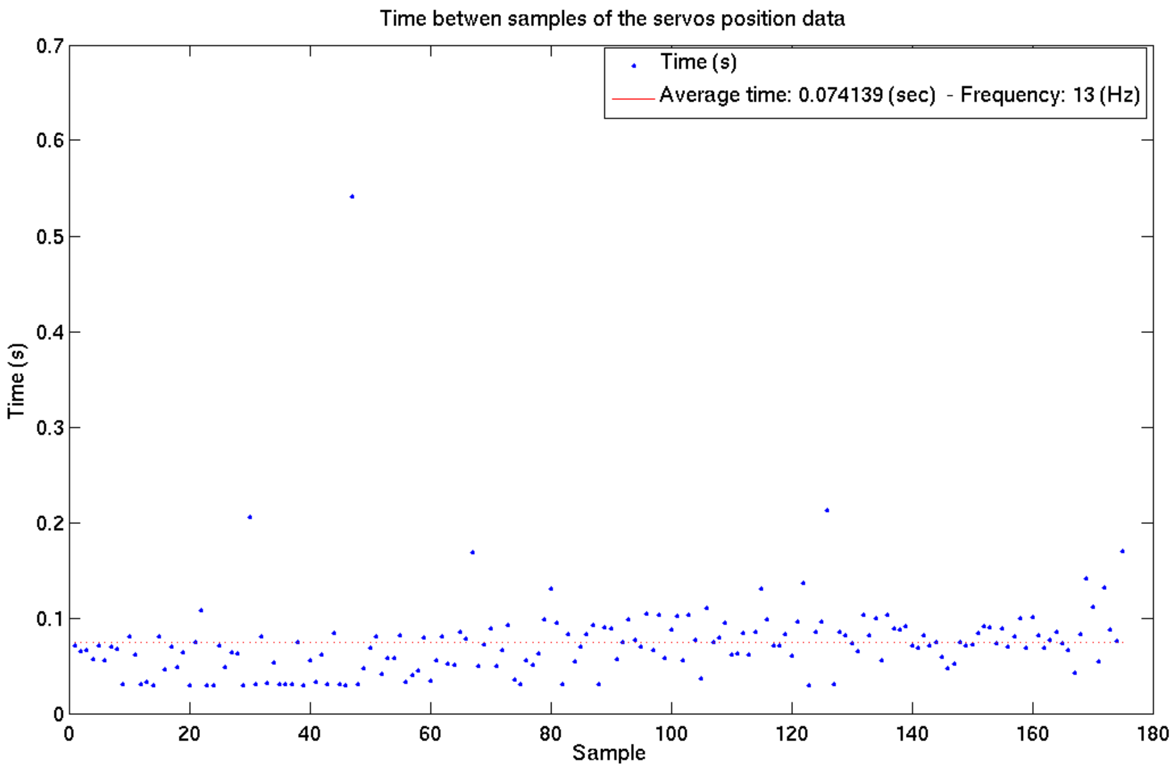
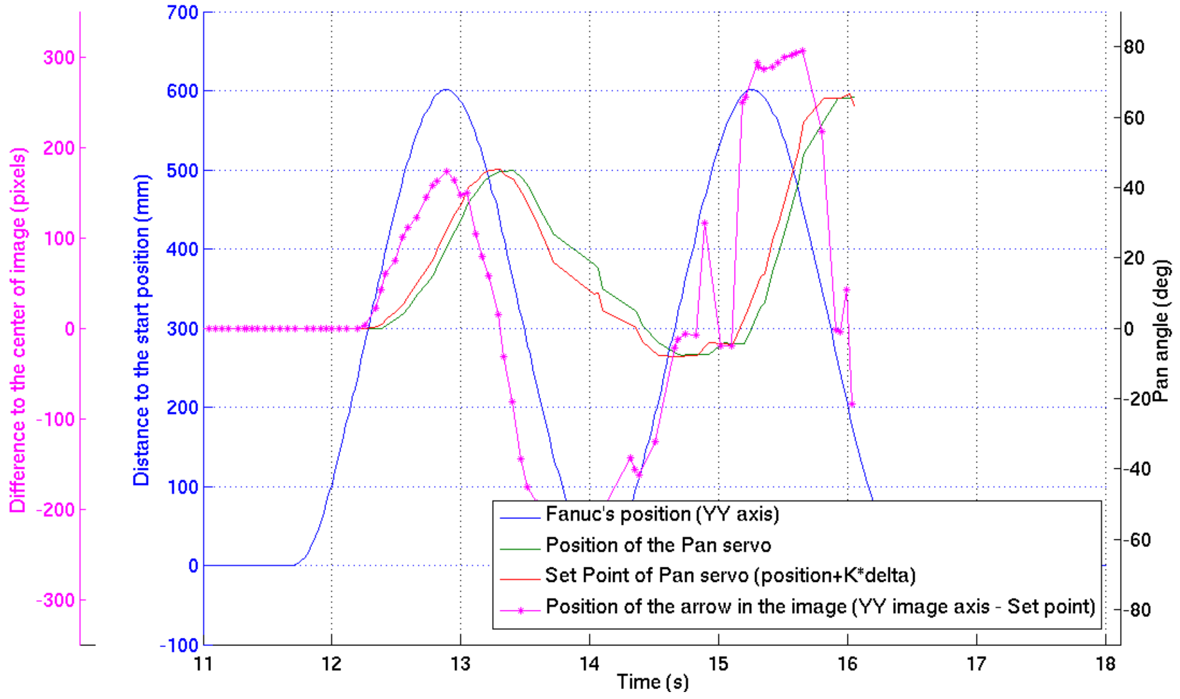


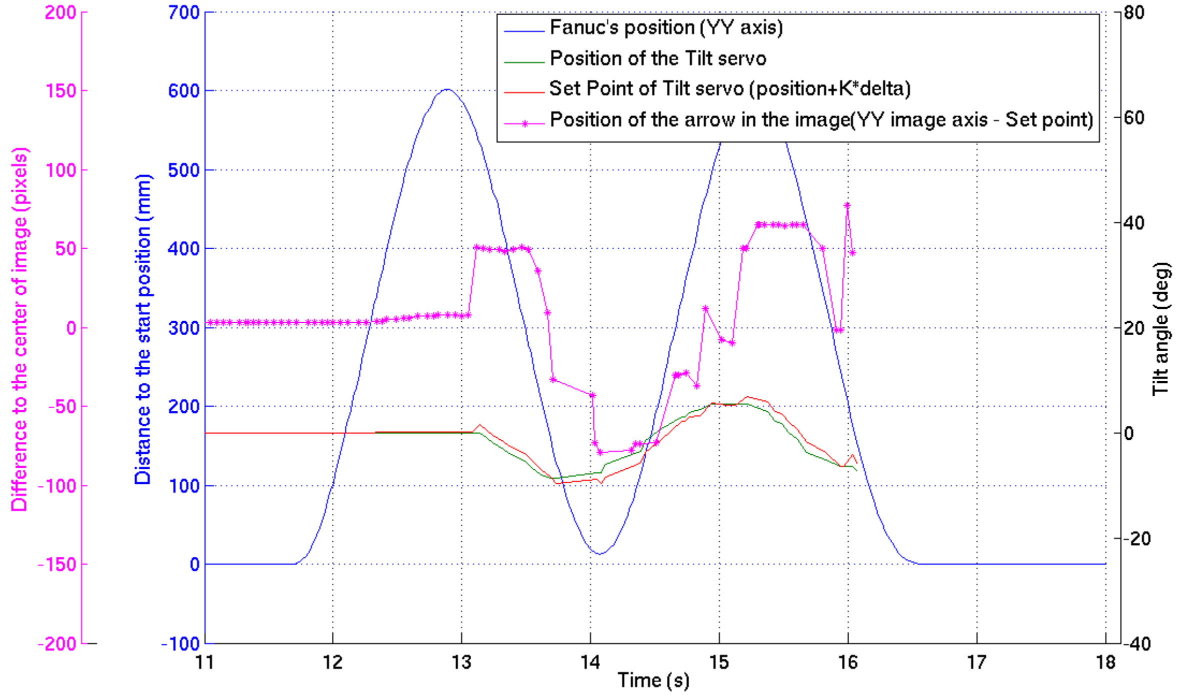
Figure 5.12: Graphical representation of the frequency of the published data by the servos controller node in the experience 5).

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 5.11, it should be noted that, given the fact that the servomotors react to the computer vision input and they did not received any instruction during that period.

The following Figure 5.13 presents an example of an experiment in which the target was lost by the PTU's tracking system.



a)

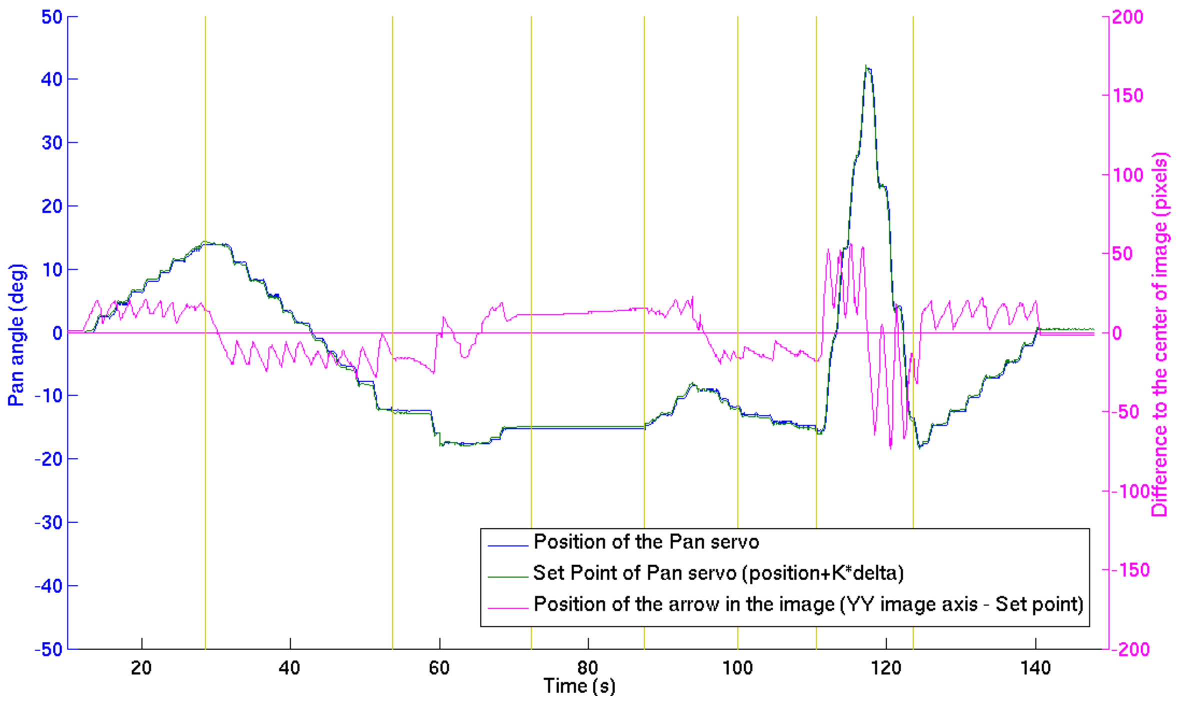


b)

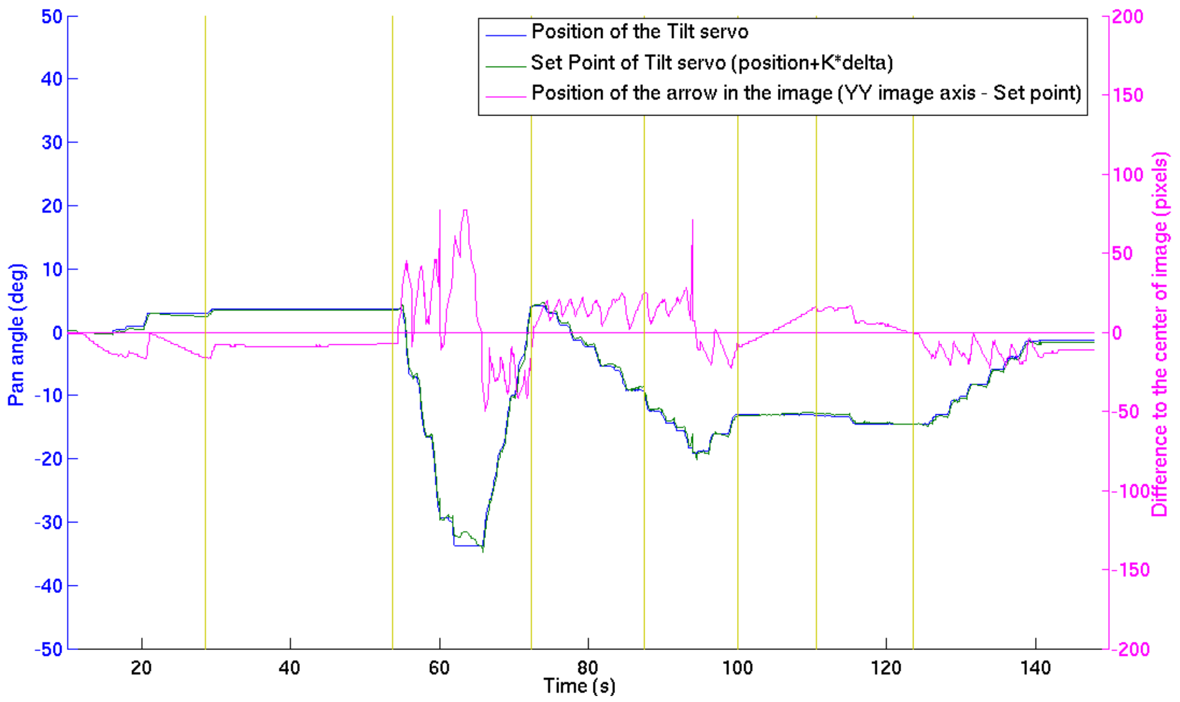
Figure 5.13: Example of a failed experiment (the target was lost by the PTU) with a position tracking control and the Fanuc robot movement profile B. The graphic a) suits for the pan component of the PTU and b) for the tilt.

The Figure 5.13 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).

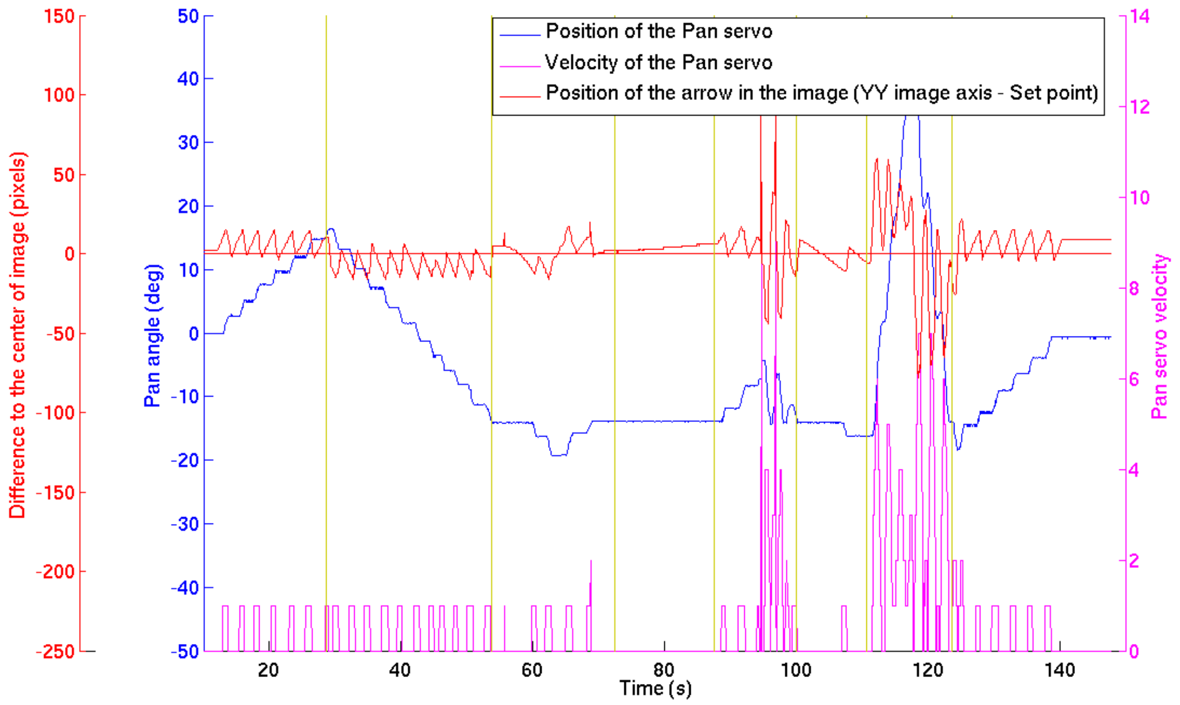


a)

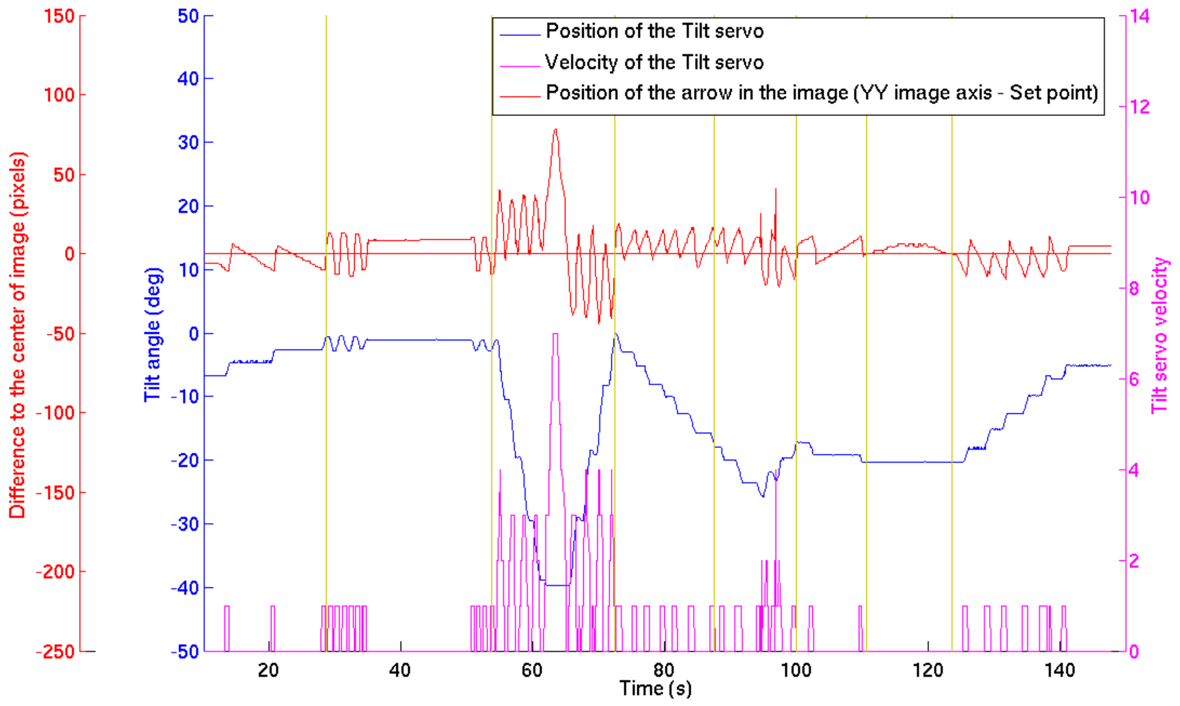


b)

Figure 5.14: Illustration of the performance of the position tracking control for the experiment 8). The graphic a) suits for the pan component of the PTU and b) for the tilt.



a)



b)

Figure 5.15: Illustration of the performance of the velocity tracking control for the experiment 4). The graphic a) suits for the pan component of the PTU and b) for the tilt.

The Figures 5.14 and 5.15 demonstrates the expected distinctions between each segment of the movement of the Fanuc robot. Depending on the movement at any given time (see figure 5.8), there is one and/or another joint of the PTU that is reacting. In order to help the visualization, besides the vertical yellow lines, it was also draw 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.

5.3 PTU Simulator

The previous described MATLAB simulator on section 4.3 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 5.2. 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, witch 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.

The Figure 5.16 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 $0,559\,(meters/second)$ 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 them selves.

Despite the *tilt* data of the experiment exposed in the Figure 5.16 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 the Figure 5.17 based on the simulated movement presented one the previous Figure 5.16 (in degrees).

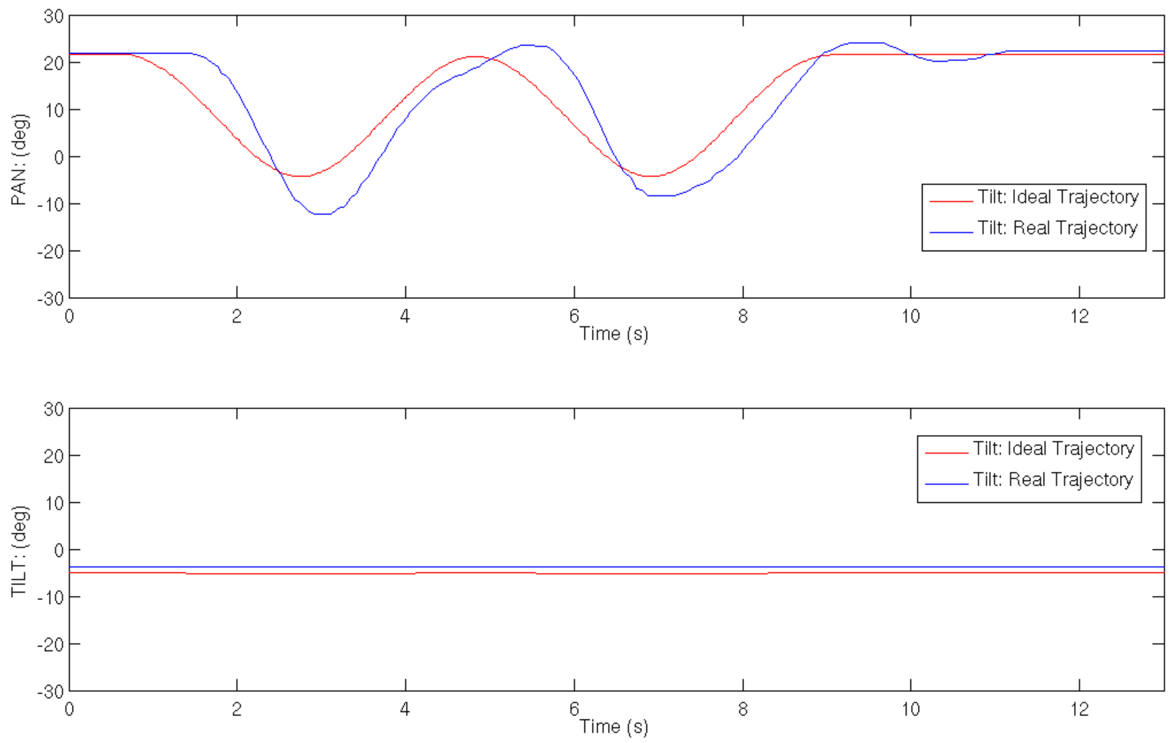


Figure 5.16: MATLAB Simulation of the control by velocity experiment with a linear movement B and 0.559(m/s) of maximum Fanuc velocity.

Regarding the Figure 5.17, it represents the difference between the ideal SP calculated be the MATLAB simulator which includes the kinematics equations of the PUT and the real position of the servomotors for both *pan* and *tilt* components.

By analyzing the Figure 5.17, 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 the section 4.2on 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, the Figure 5.18 presents the output graphics of the error of the target centroid witch is projected on the image (in meters).

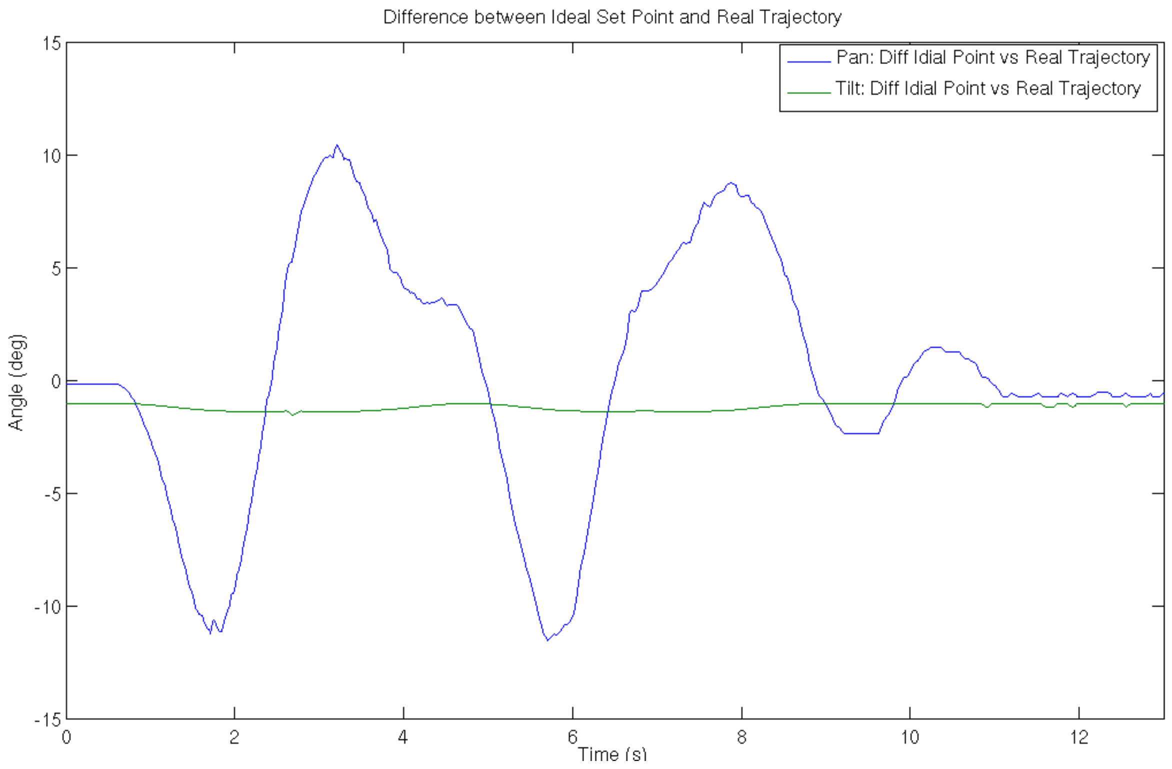


Figure 5.17: Introduced error of the PTU control system on movement.

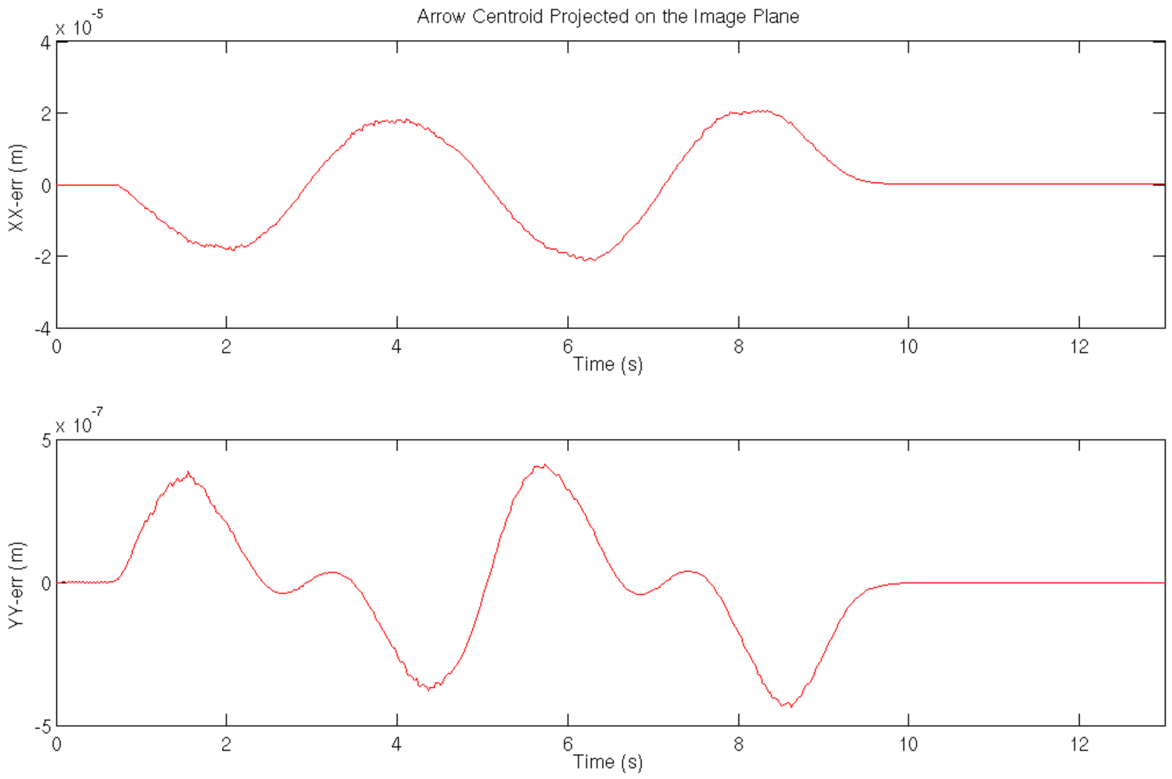


Figure 5.18: Error of the target's centroid projected on image plane.

The Table 5.8 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 5.8: 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 5.8 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 the Figure 5.18 for the error on the image plane.

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, the Figure 5.19 is used to illustrate a movement of the PTU on a given moment exactly when the camera captures the target on the image[5]. The Figure also illustrates the position where the camera captures the image within the target on the center of it ($C2$).

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

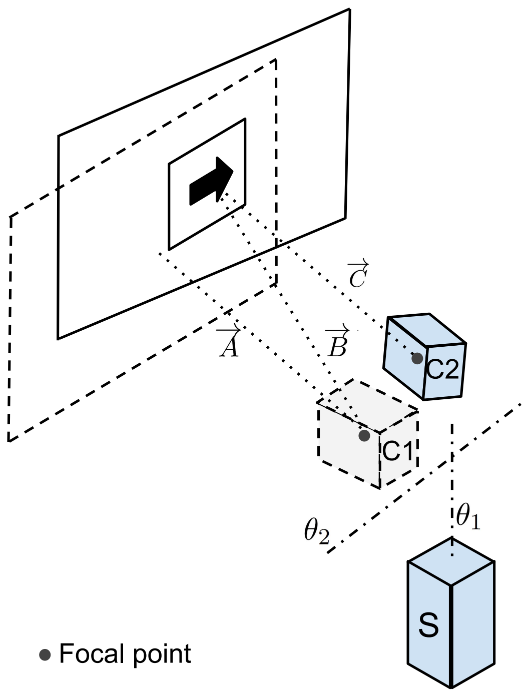


Figure 5.19: Illustration of a PTU's movement from an given point gathering the target's direction, towards a position where the camera is faced with the target in the center of the image.

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 PUT only has 2 DOF; for a specific position of its end-effector[6], 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.

[6]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[7].

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. 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. The Figure 5.20 illustrates the proposal of the author based on the gyroscope principle on a) inspired on the mechanical structure presented on b).

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.

Regarding the upgraded PTU's mechanical structure, considering the large number of experiments conducted during the development of the present dissertation, it has

[7](see section 4.1.2)

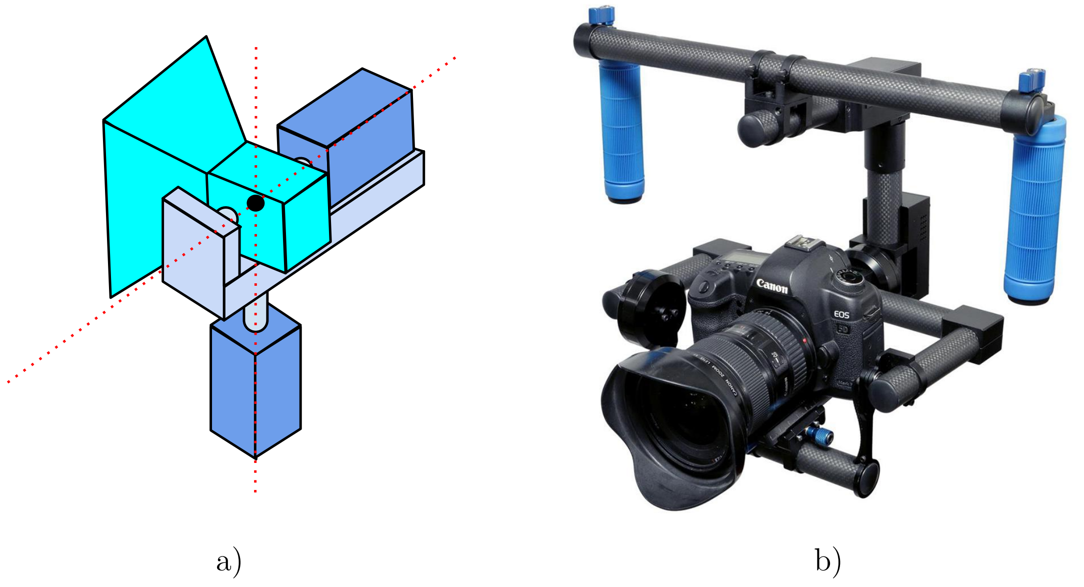


Figure 5.20: a) New PTU's mechanical structure proposal, b) 2-Axis Digital Gyro Stabilizer made by the company Adorama [1].

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.

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 the PHUA's neck and it did not presented any problems during the experimental tests. This infrastructure offers the possibility to be easily adapted to the use of 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 both OpenCV, ViSP and ROS was successfully accomplished working now together in this project. Besides the use of the 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 any 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 the real robot environment, without artificial landmarks.

It was developed an 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.

The control of the servomotors is made with a simple proportional control, along with a more powerful computational capabilities must be improved, 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.

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 the visual feedback for the humanoid.

6.1 Future Work

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

- Make the integration of this work with the existent balance control system;
- Implement an autonomous target selection, may follow the line that the flowing references suggest [22, 10, 38];
- Modulate the neck structure in CATIA® software, to be accorded to the other parts of the PHUA which are drawn on this software;
- Consider to improve the neck control system with a Proportional-Integral-Derivative controller (PID);
- Implement a ROS service protocol for the visual tracking system request the target position only when necessary;
- Test the performance of the developed system with a better computer processor.

References

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

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

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

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

[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.

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

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

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

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

[10] 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.

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

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

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

[14] 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.

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

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

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

[18] 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.

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

[20] 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.

[21] Russell D Fernald. Evolving eyes. *The International journal of developmental biology*, 48(8-9):701–5, January 2004.

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

[23] 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.

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

[25] 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.

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

[27] 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.

[28] 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.

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

[30] 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.

[31] 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.

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

[33] 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.

[34] P. McKerrow. *Introduction to Robotics.* ACM Press frontier series. Addison-Wesley Publishing Company, 1991.

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

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

[37] 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.

[38] 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.

[39] 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.

[40] 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.

[41] Alexandra S. Pollock, Brian R. Durward, Philip J. Rowe, and John P. Paul. What is balance? *Clinical Rehabilitation*, 14(4):402–406, April 2000.

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

[43] 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.

[44] 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.

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

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

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

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

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

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

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

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

[53] 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.

[54] 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.

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

[56] Sony. Qrio. `http://www.sony.net/`. consulted at 12-8-2014.

[57] 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.

[58] S. Ushida and K. Deguchi. *Vision-based Motion Control of a Biped Robot Using 2 DOF Gaze Control Structure.* INTECH Open Access Publisher, 2007.

[59] 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.

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

[61] 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.

[62] 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.

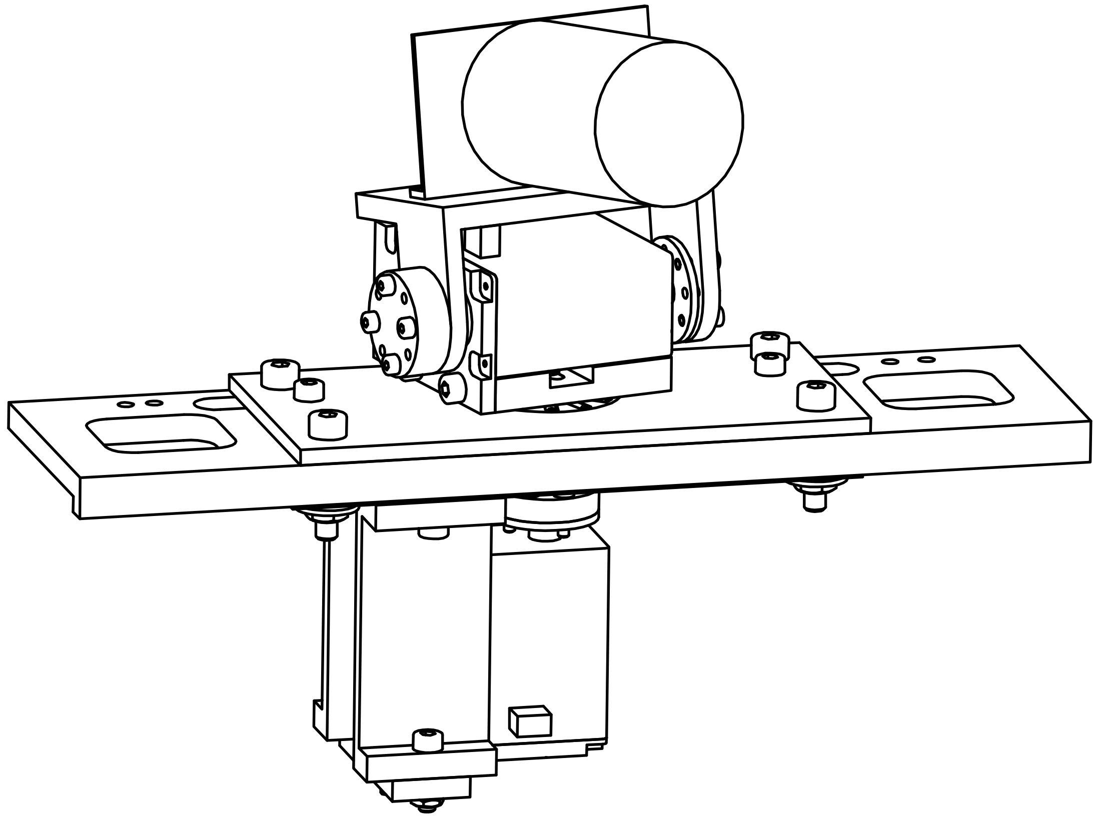
[63] Craig R. White, Norman Day, Patrick J. Butler, and Graham R. Martin. Vision and foraging in cormorants: More like herons than hawks? *PLoS ONE*, 2(7), July 2007.

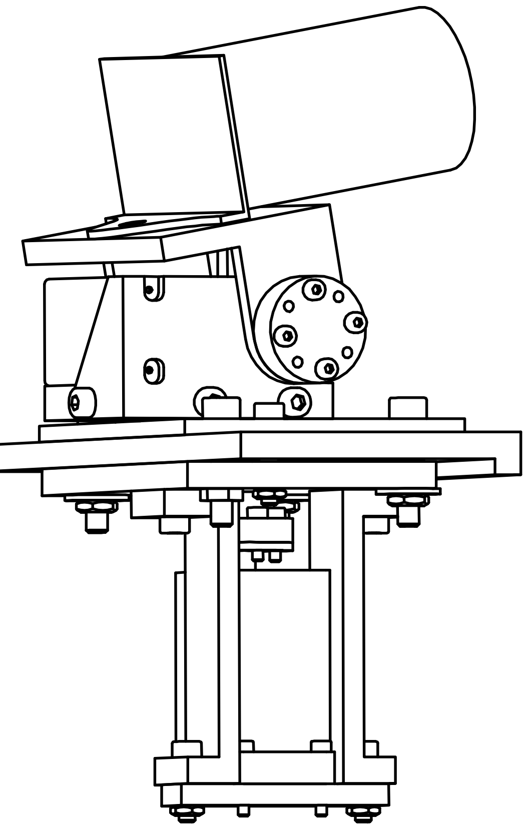
[64] Wikipedia. `http://en.wikipedia.org/`. consulted at 14-8-2014.

[65] 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.

Appendix A

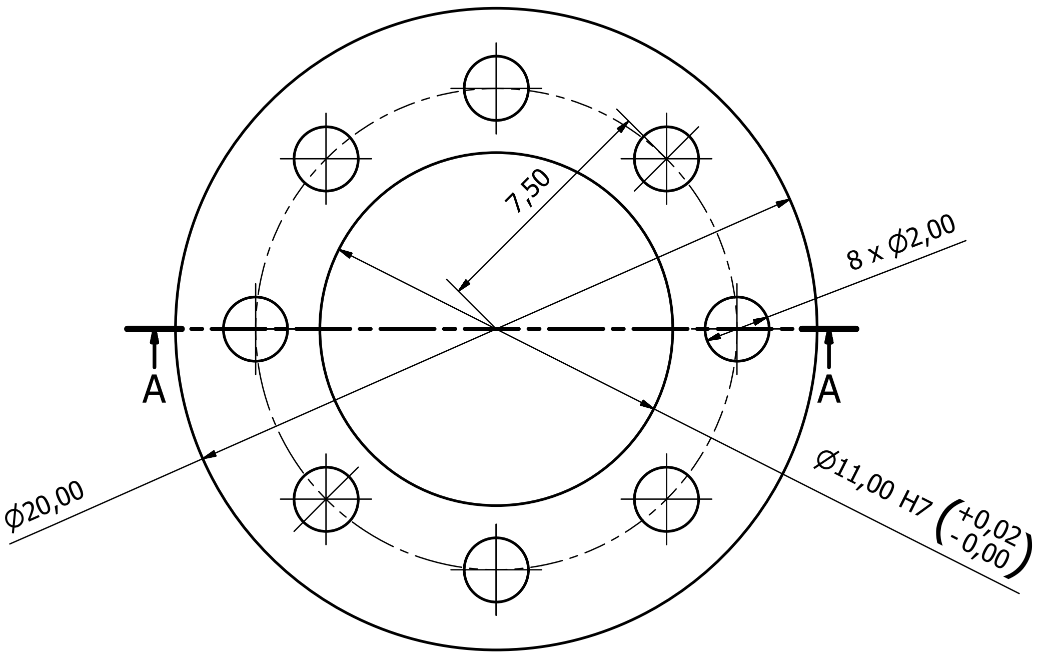
Mechanical Drawings



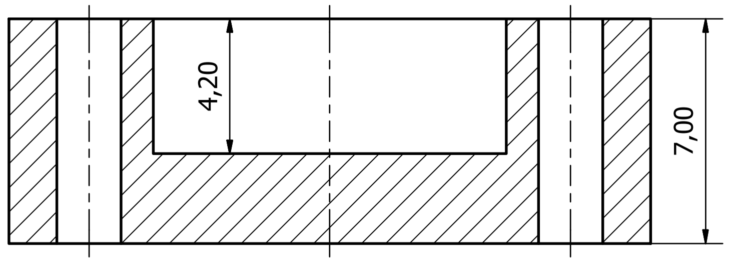


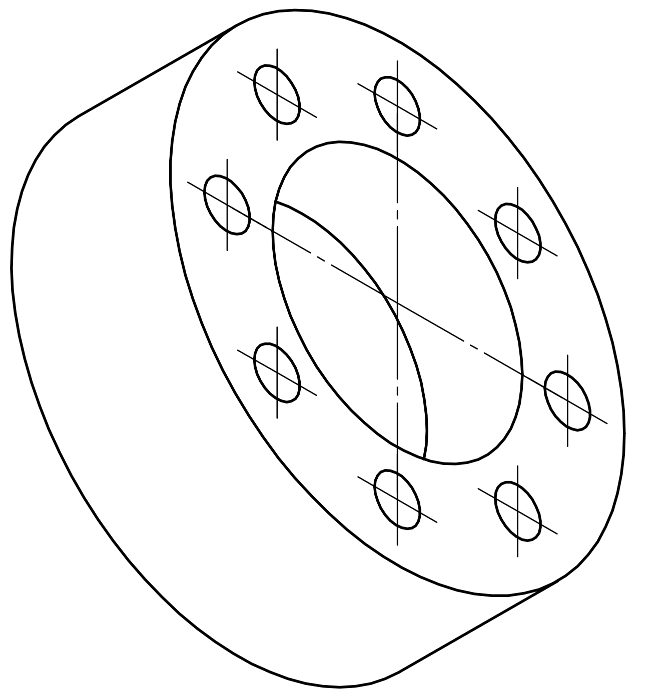
Projetado por	Verificado por	Aprovado por	Data	Data
César Sousa 49649				02-12-2013

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

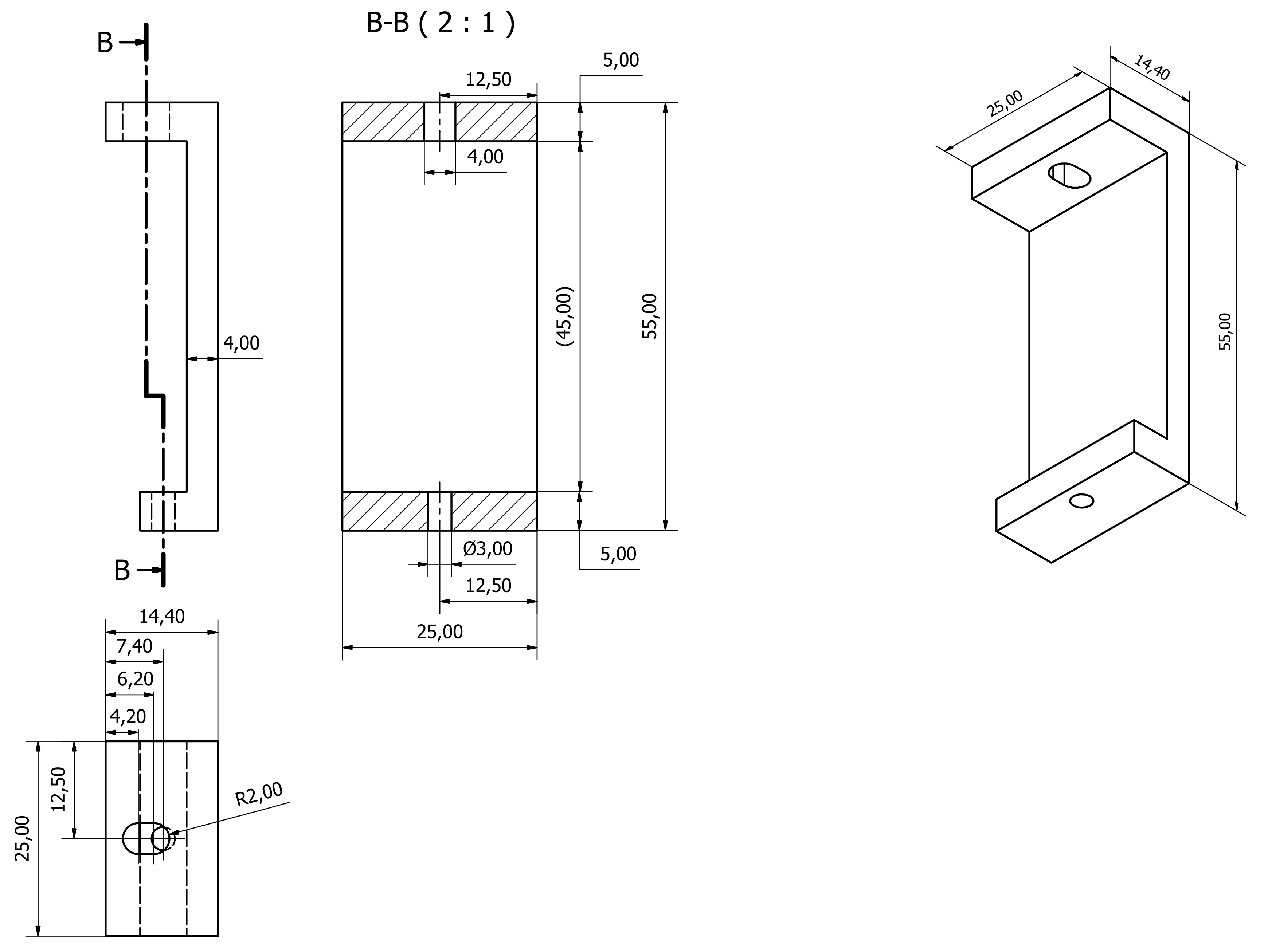

7,50
8 x Ø2,00
A
A
Ø20,00
Ø11,00 H7 $\left(\begin{smallmatrix}+0,02\\-0,00\end{smallmatrix}\right)$

A-A (6 : 1)

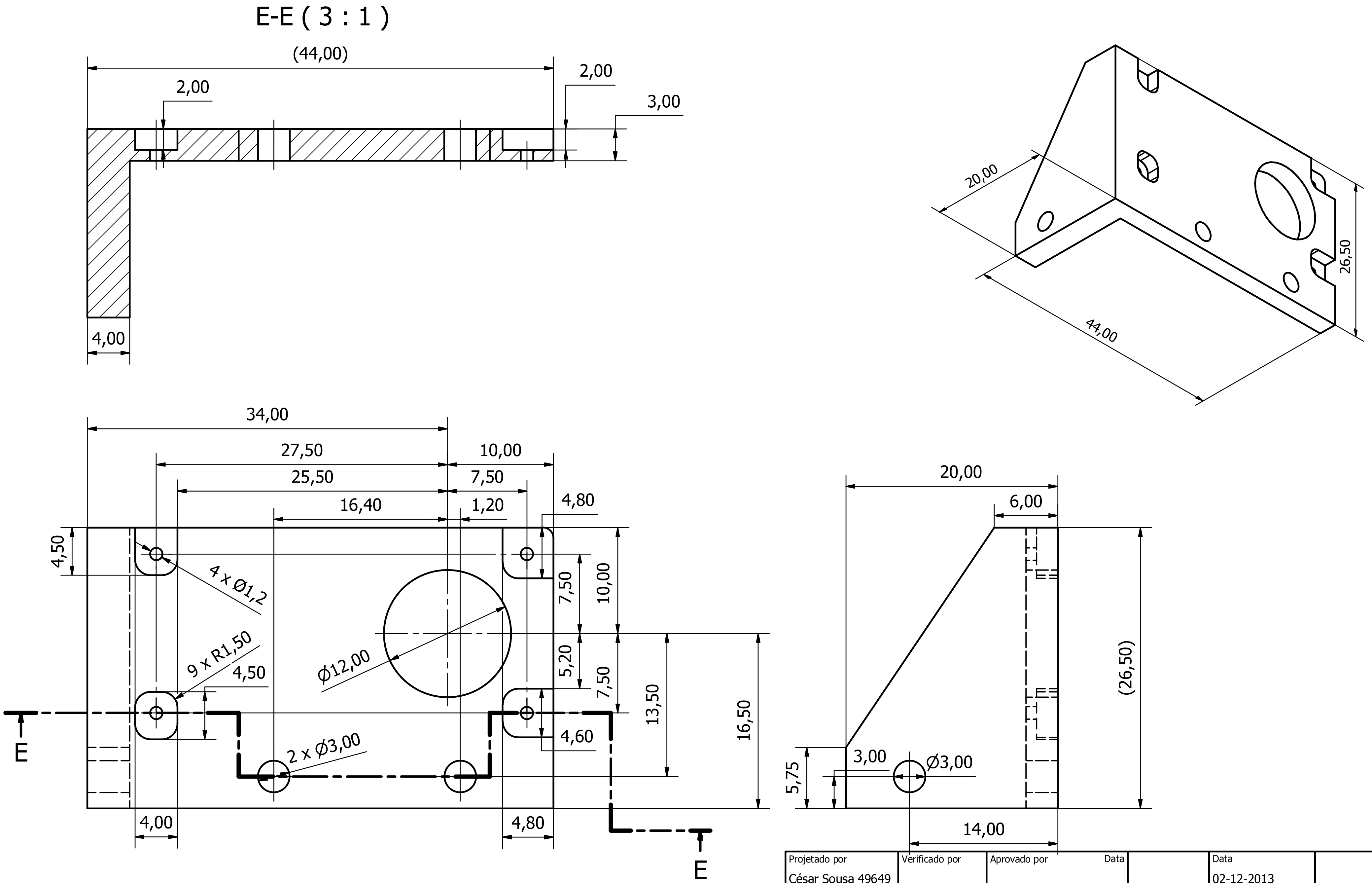




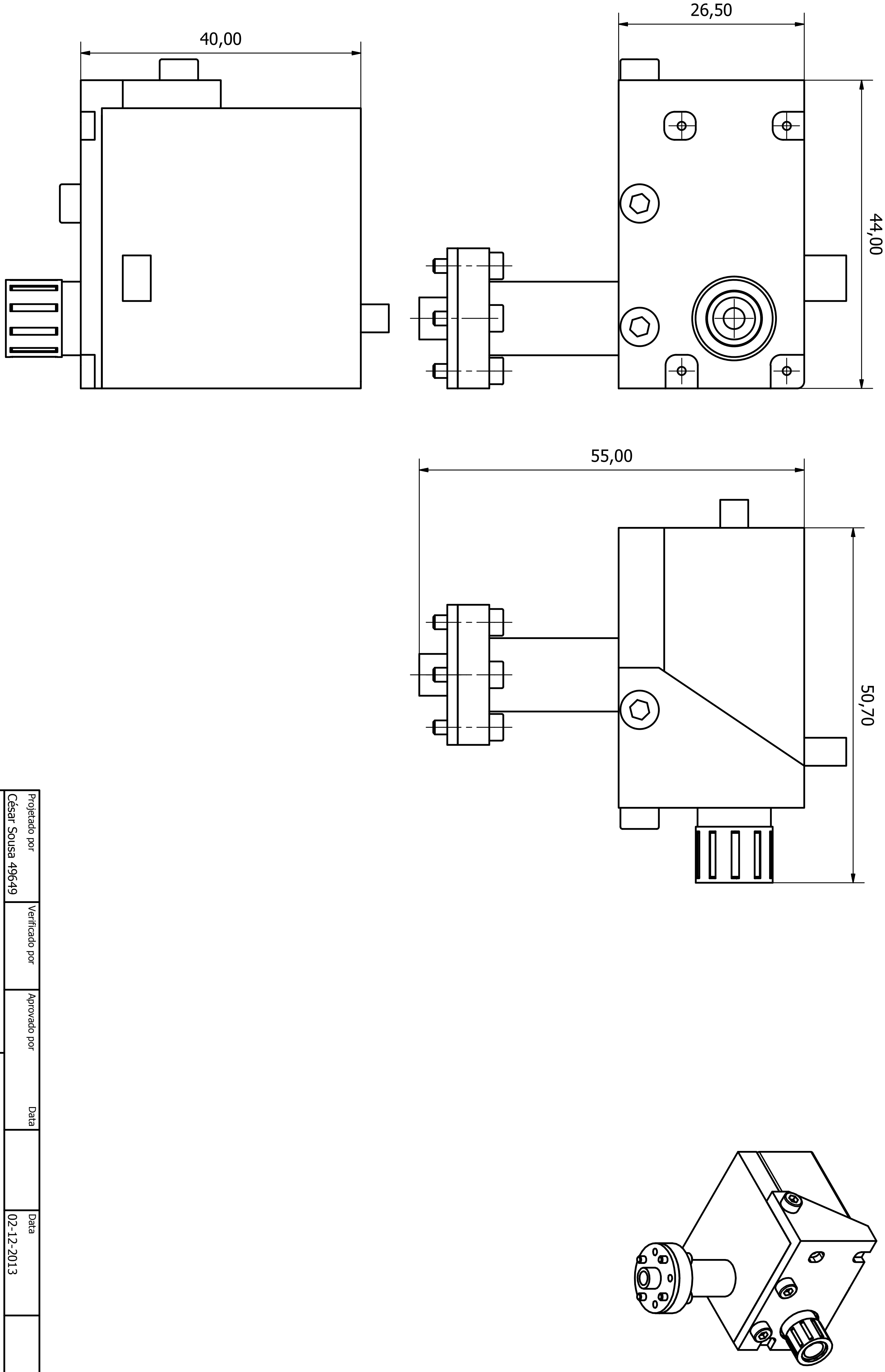
Projetado por César Sousa 49649	Verificado por	Aprovado por	Data		Data 02-12-2013
Projeto Humanoid UA- Prof. Vitor Santos e Filipe Silva		1 Peça			
		draw_caixa_rolamento_v1		Edição	Folha 1 / 1



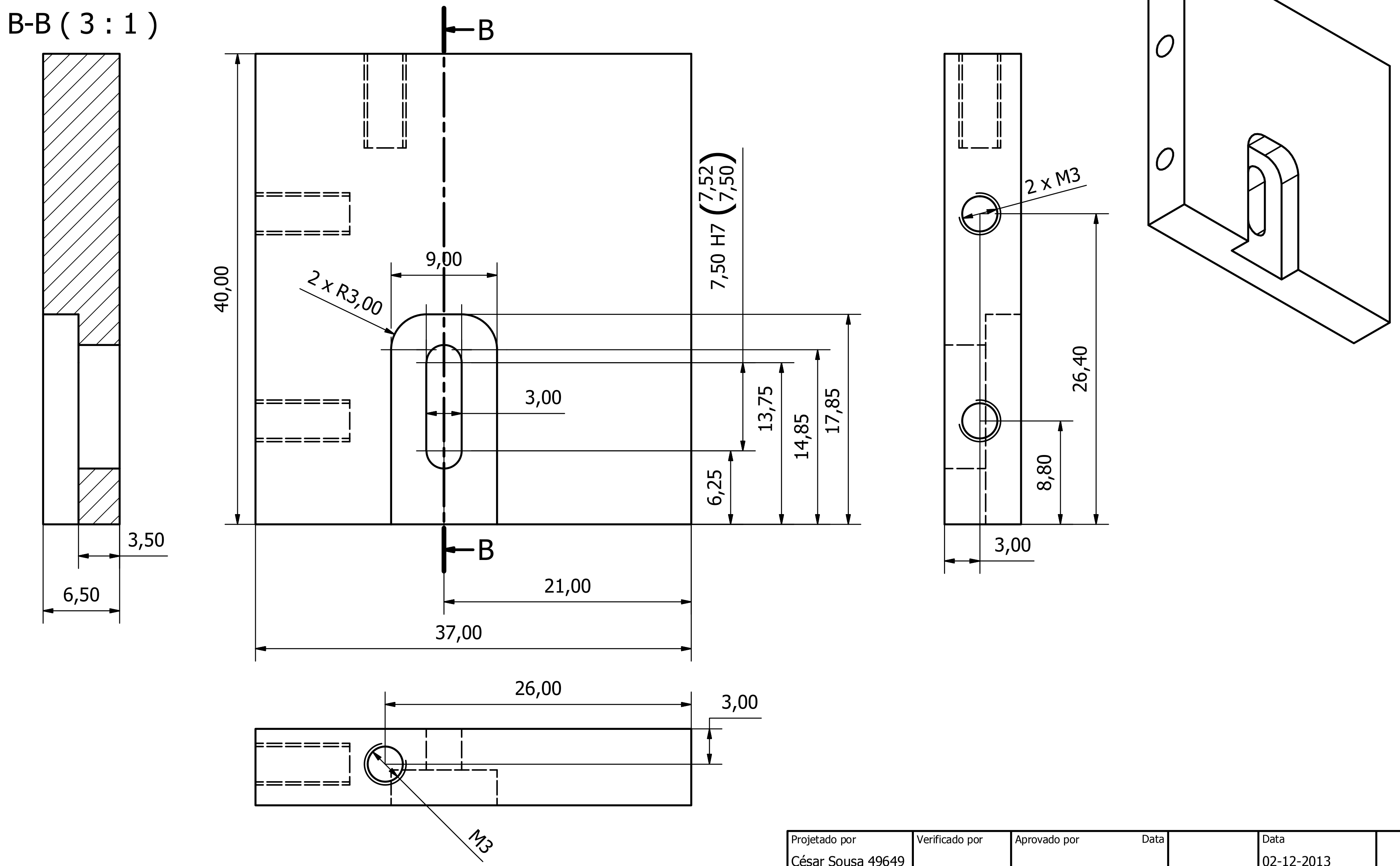
Projetado por César Sousa 49649	Verificado por	Aprovado por	Data	Data 02-12-2013	
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva		2 PEÇAS			
		draw_suporte_servo_baixo_v2		Edição	Folha 1 / 1



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_chapa_servo2_v2		Edição	Folha 1 / 1	

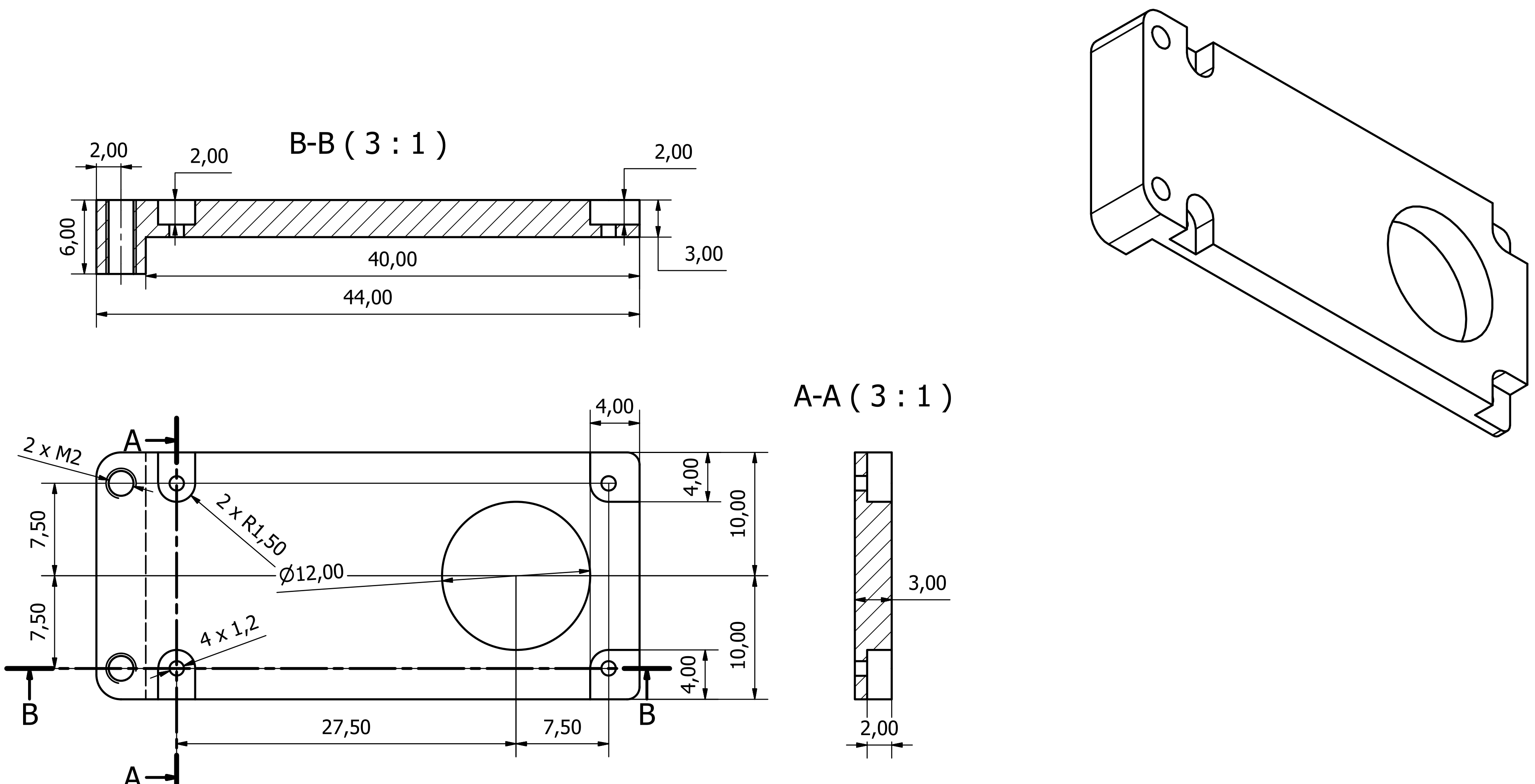


Projetado por	Verificado por	Aprovado por	Data	Data
César Sousa 49649				02-12-2013
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva			1 peça	
draw_mont_meio-1			Edição	Folha 1 / 1

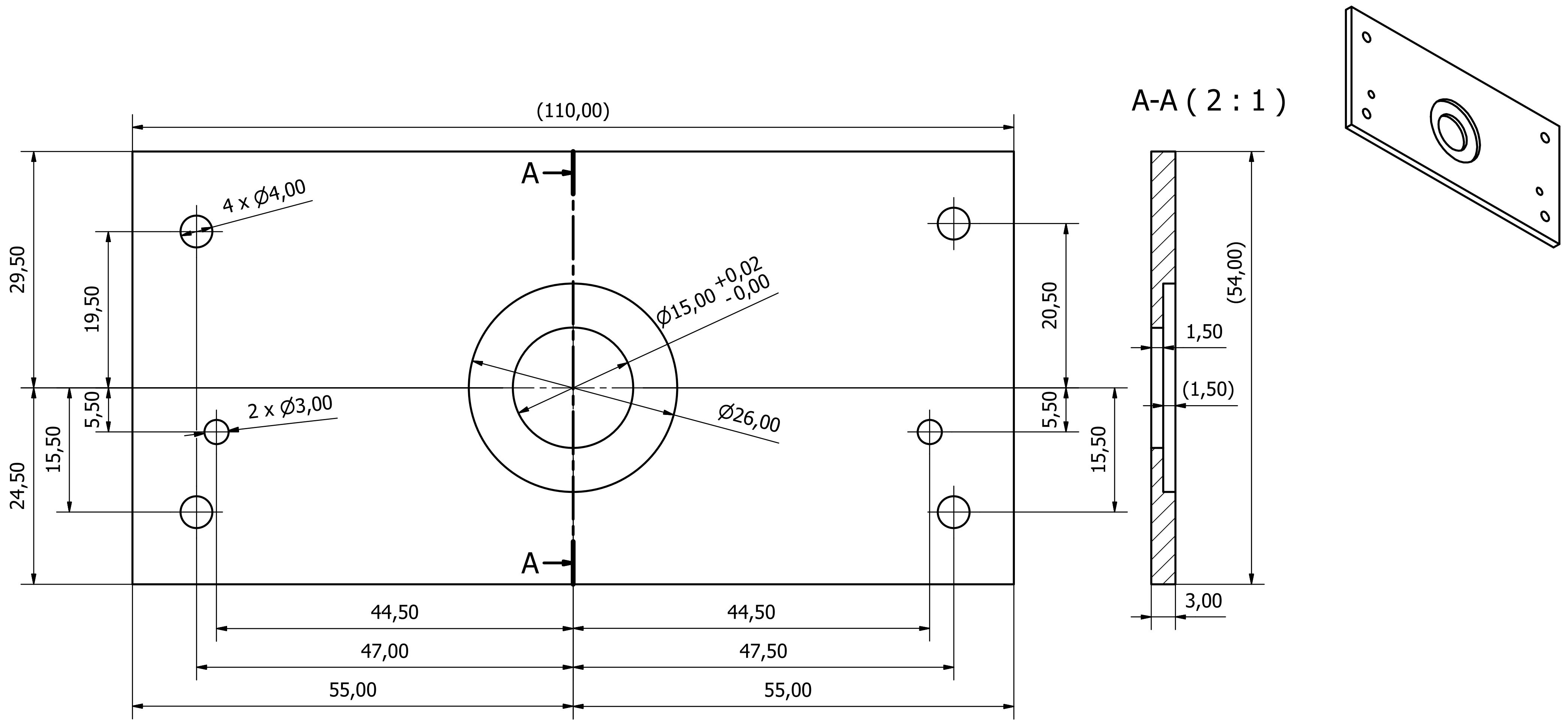


Projetado por	Verificado por	Aprovado por	Data	Data	
César Sousa 49649				02-12-2013	

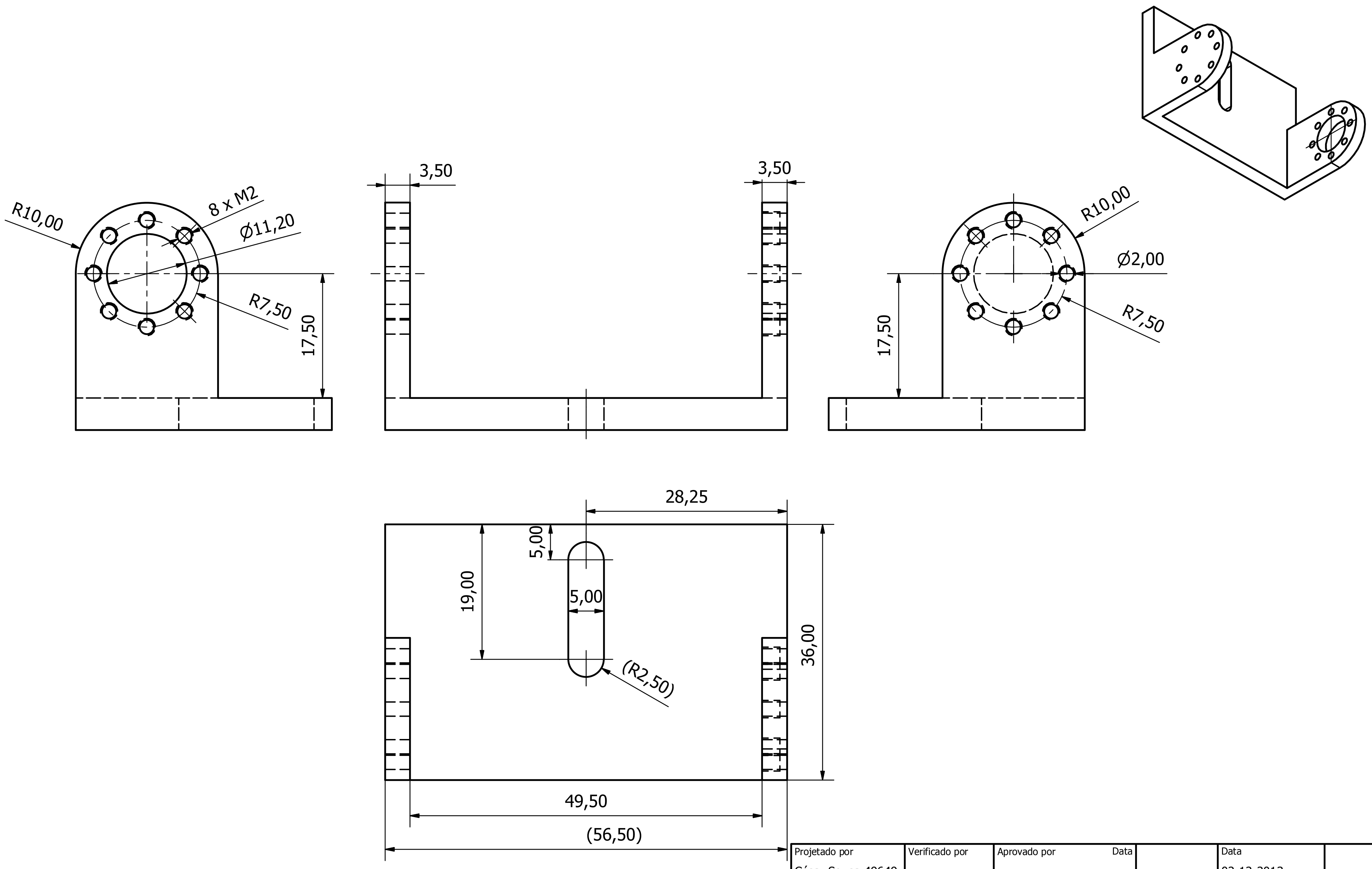
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça		
	draw_suporte_servo_v5	Edição	Folha 1 / 1



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_chapa_servo_v2		Edição	Folha 1 / 1

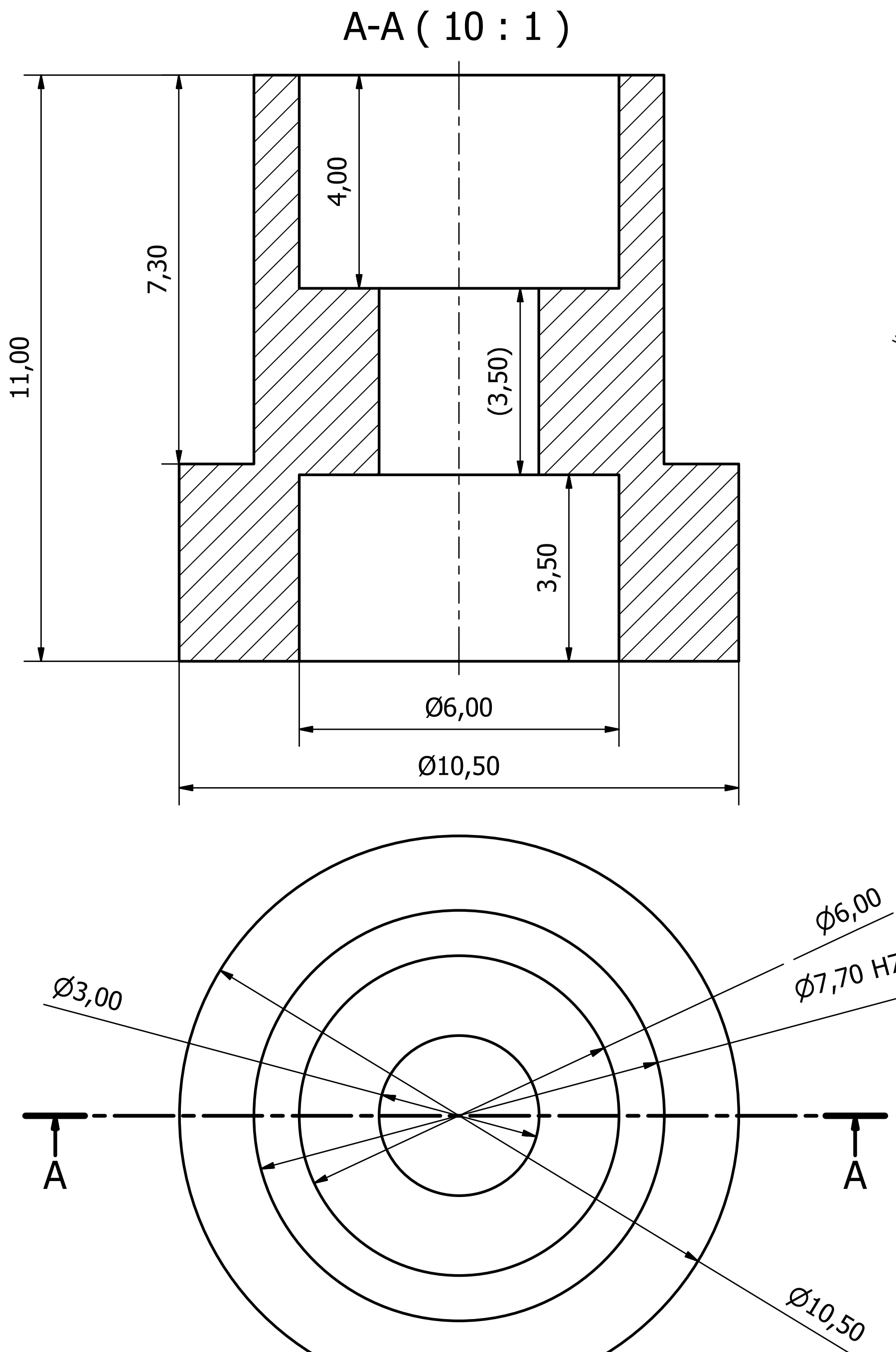


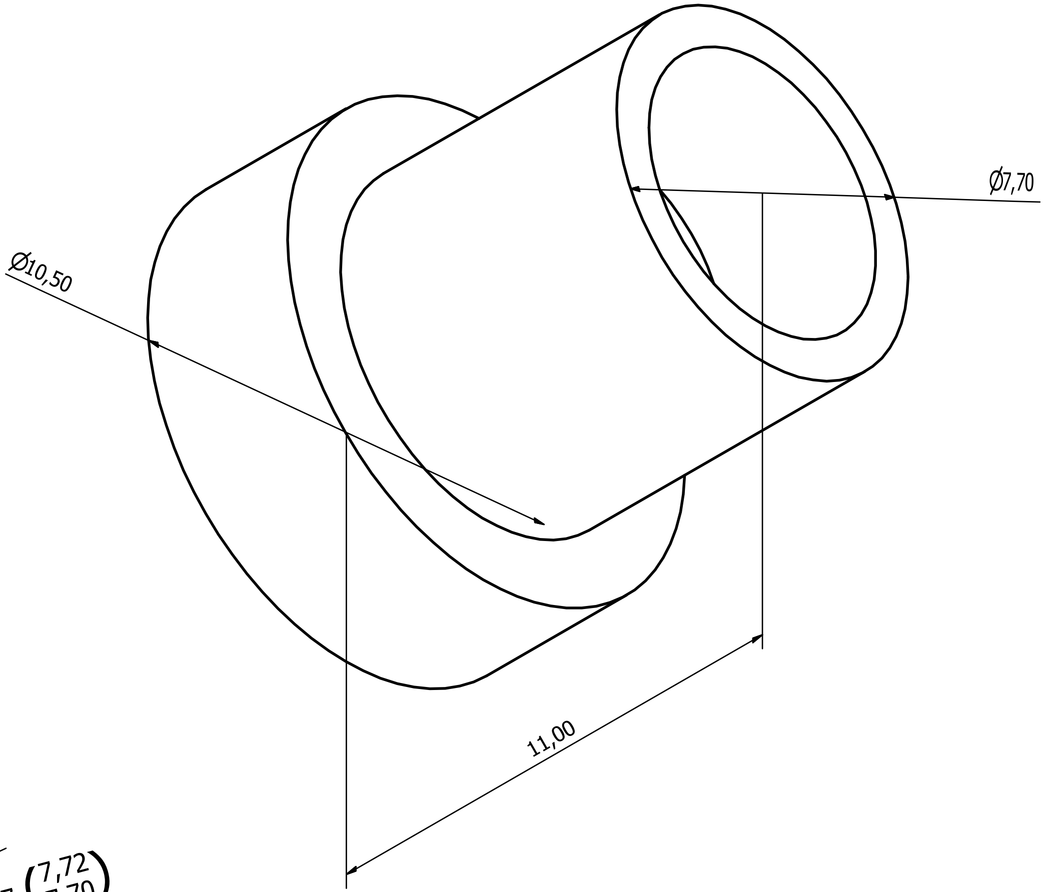
Projetado por César Sousa 49649	Verificado por	Aprovado por	Data	Data 02-12-2013	
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva		1 peça			
		draw_suporte_rolamento-1		Edição	Folha 1 / 1



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



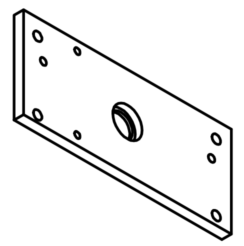


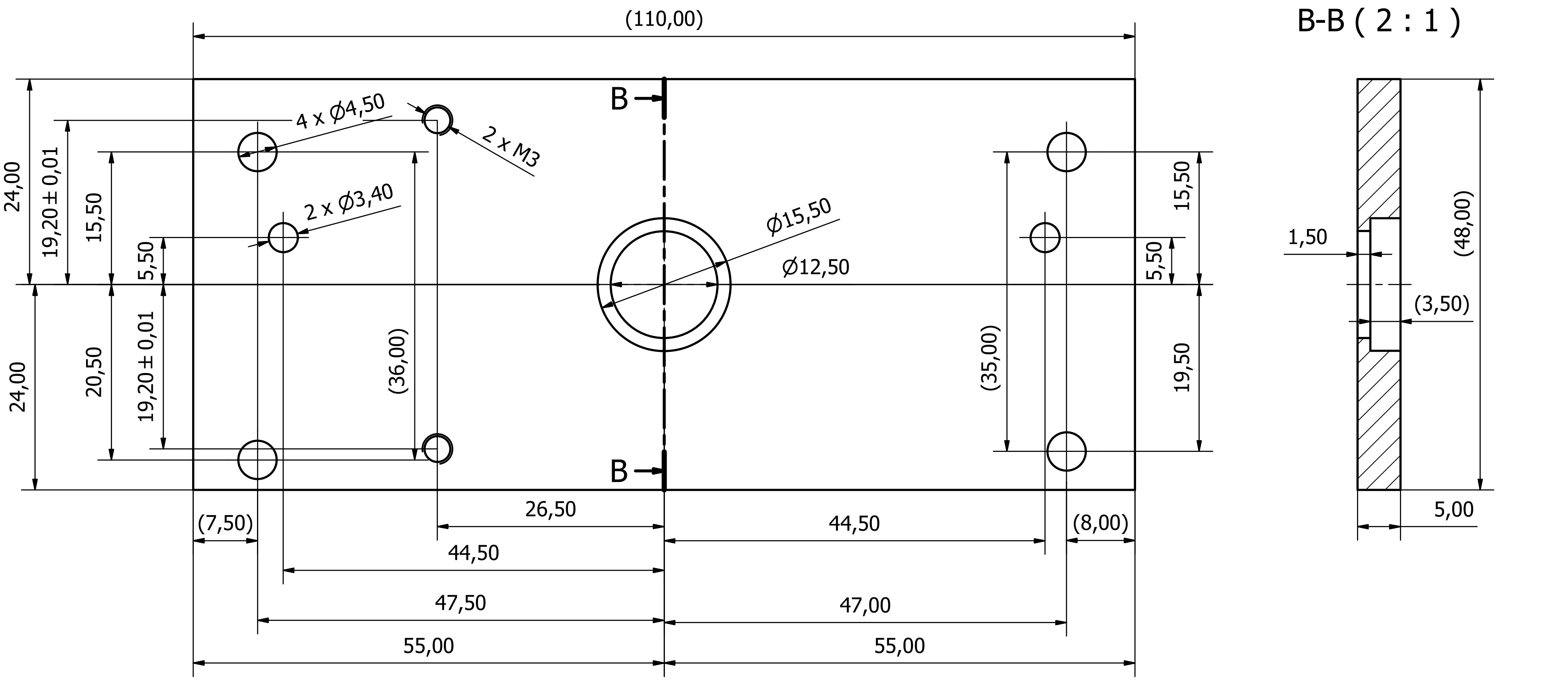
Projetado por	Verificado por	Aprovado por	Data	Data
César Sousa 49649				02-12-2013

Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva

1 peça

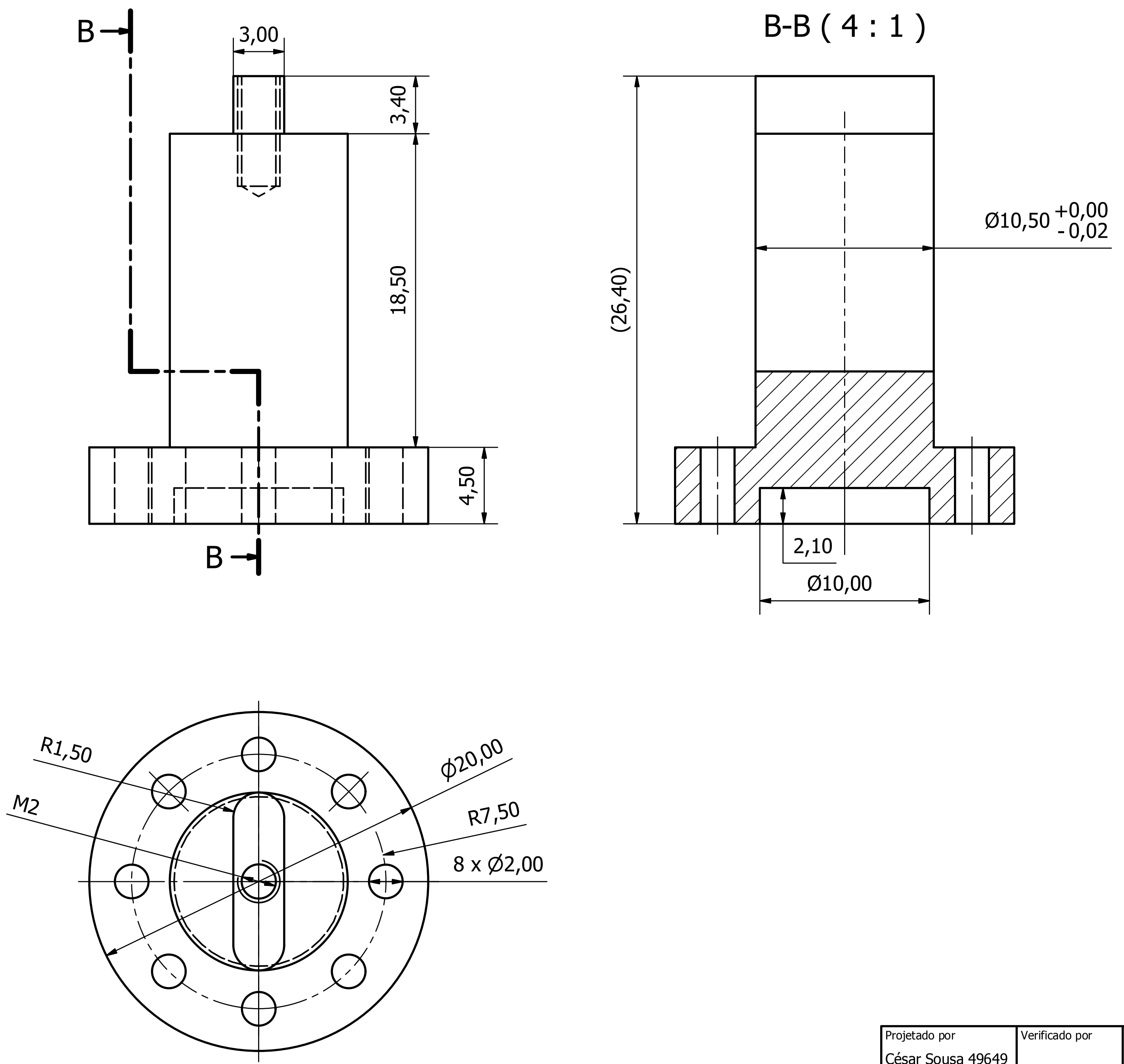
	Edição	Folha
draw_guia_servo_v0		1 / 1

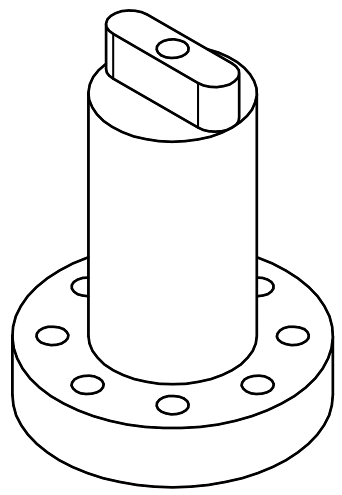




Projetado por	Verificado por	Aprovado por	Data	Data	
César Sousa 49649				02-12-2013	

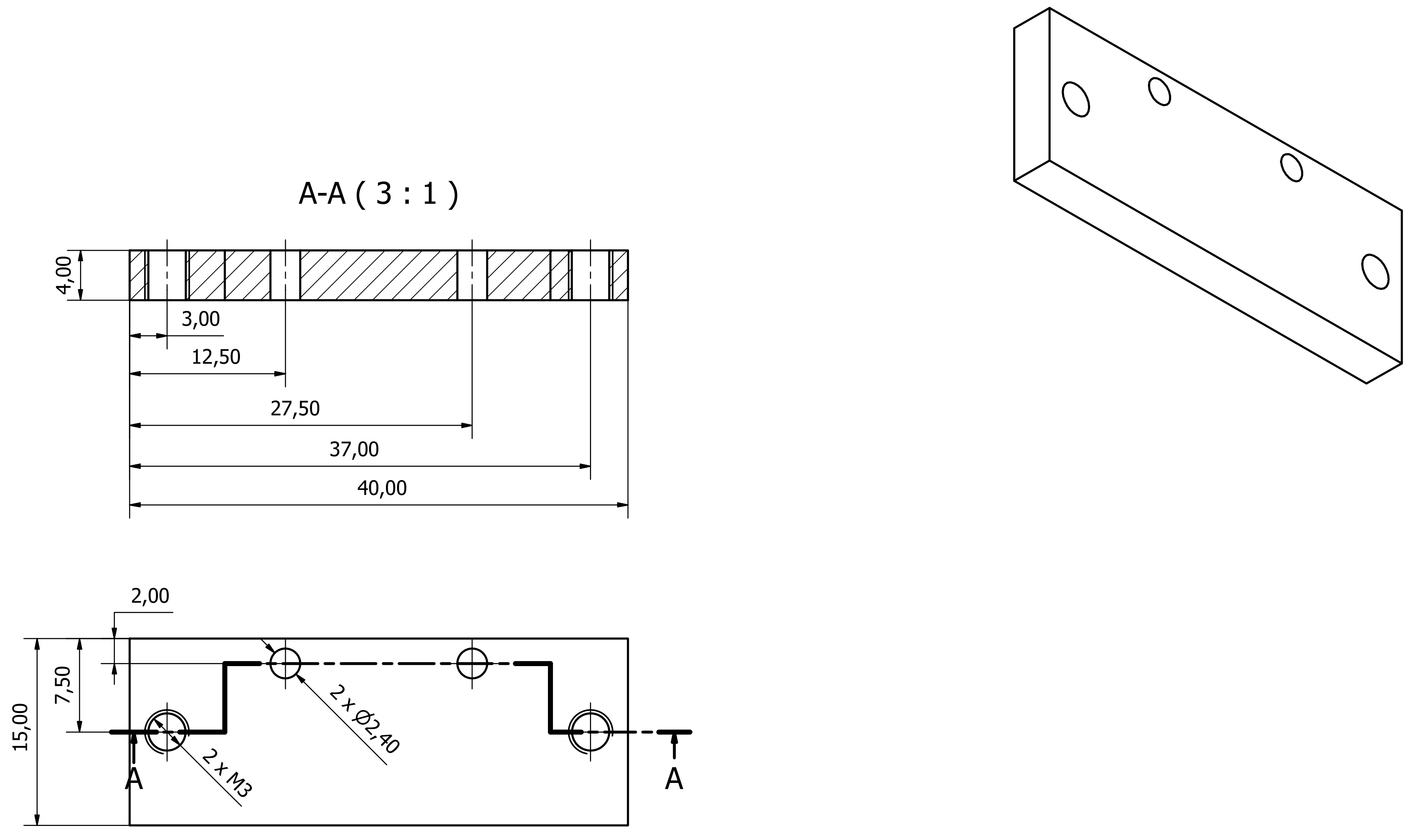
Projeto Humanoid UA - Profs. Vitor Santos e Filipe Silva	1 peça
	draw_suporte_rolamento_guia_v0 — Edição — Folha 1 / 1





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_ligacao_entre_servos_v4	Edição	Folha 1 / 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_suporte_servo_baixo2_v2	Edição	Folha 1 / 1

Appendix B

Find Target Algorithm

Algorithm B.1 Track arrow pseudo-code

```
initialize receive_new_point equal false
initialize global_point

main
  initialize ROS node stuff
  initialize arrow_position and camera_image subscriber
  initialize visp image display
  get the center of the image
  initialize blob tracking parameters
  set lost_arrow equal true
  while true
    get frame
    if receive_new_point
      receive_new_point equal false
      lost_arrow equal false
      initialize blob tracking with the point arrow_position
    if lost_arrow equal false
      track blob
      get blob center
      publish(calculate_command(blob center))
      refresh image
    if caught exception
      lost_arrow equal true
```

```
Point receiver callback
  update global point
  receive_new_point equal true

calculate_command(blob)
    calculate the difference between the blob's center and image center
    return the point
```

Algorithm B.2 Servos position controller pseudo-code

```
receive_new_point equal false
initialize global_point
initialize servos connection

main
  initialize the last_position vars with the center position
  initialize ROS node stuff
  initialize set point subscriber
  initialize thread

Point receiver callback
  update global point
  receive_new_point equal true

setServosPos(servo_id,max_position,min_position,set_point)
  set_point less then min_position?
    set servo position to min_position
    return min_position
  set_point greater then max_position?
    set servo position to max_position
    return max_position
  else
    set servo position to set_point
    return set_point

tread
  while true
    received_new_point?
      received_new_point = false
      velocity control?
        calculate velocity
        velocity valid?
          tilt velocity equal 0?
            get tilt servo position and set velocity to 0
            send tilt servo to same position
          else
            get tilt servo position and set velocity
            direction down?
              node is initializing or the direction changed?
                save this new direction
                set tilt servo to min position
            else
              node is initializing or the direction changed?
                save this new direction
                set tilt servo to max position
          pan velocity equal 0?
            get pan servo position and set velocity to 0
            send pan servo to same position
          else
            get pan servo position and set velocity
            direction right?
              node is initializing or the direction changed?
                save this new direction
                set pan servo to min position
            else
              node is initializing or the direction changed?
                save this new direction
                set pan servo to max position
      else[a]
        get pan servo position and set to max velocity
        calculate new position
        setServosPos(new position,max and min position, pan servo id)
        get tilt servo position and set to max velocity
        calculate new position
        setServosPos(new position,max and min position, tilt servo id)
```

[a]position control