OBJECT MANIPULATION BY A HUMANOID ROBOT VIA SINGLE CAMERA POSE ESTIMATION

by
ŞEFİK EMRE ESKİMEZ

Submitted to the Graduate School of Engineering and Natural Sciences in partial fulfillment of the requirements for the degree of Master of Science

Sabancı University
August 2013

OBJECT MANIPULATION BY A HUMANOID ROBOT VIA SINGLE CAMERA POSE ESTIMATION

APPROVED BY

Assoc. Prof. Dr. Kemalettin ERBATUR ……………………………………….
(Thesis Advisor)

Assoc. Prof. Dr. Volkan PATOĞLU ……………………………………….

Assoc. Prof. Dr. Albert LEVI ……………………………………….

Assoc. Prof. Dr. Meriç ÖZCAN ……………………………………….

Assoc. Prof. Dr. Hakan ERDOĞAN ……………………………………….

DATE OF APPROVAL ……………………………………….

OBJECT MANIPULATION BY A HUMANOID ROBOT VIA SINGLE CAMERA POSE ESTIMATION

Şefik Emre Eskimez

Mechatronics Engineering, M.Sc. Thesis, 2013

Thesis Supervisor: Assoc. Prof. Dr. Kemalettin ERBATUR

Keywords: Humanoid robots, object manipulation, object grasping, computer vision systems, navigation

ABSTRACT

Humanoid robots are designed to be used in daily life as assistance robots for people. They are expected to fill the jobs that require physical labor. These robots are also considered in healthcare sector. The ultimate goal in humanoid robotics is to reach a point where robots can truly communicate with people, and to be a part of labor force.

Usual daily environment of a common person contains objects with different geometric and texture features. Such objects should be easily recognized, located and manipulated by a robot when needed. These tasks require high amount of information from environment.

The Computer Vision field interests in extraction and use of visual cues for computer systems. Visual data captured with cameras contains the most of the information needed about the environment for high level tasks relative to the other sensors. Most of the high level tasks on humanoid robots require the target object to be segmented in image and located in the 3D environment. Also, the object should be kept in image so that the information about the object can be retrieved continuously. This can be achieved by gaze control schemes by using visual feedback to drive neck motors of the robot.

In this thesis an object manipulation algorithm is proposed for a humanoid robot. A white object with red square marker is used as the target object. The object is segmented by color information. Corners of the red marker is found and used for the pose estimation algorithm and gaze control. The pose information is used for navigation to the object and for the grasping action. The described algorithm is implemented on the humanoid experiment platform SURALP (**S**abanci **U**niversity **R**ese**A**rch **L**abaratory **P**latform).

İNSANSI ROBOT İLE TEK KAMERADAN ORYANTASYON HESABI YAPILARAK NESNE MANİPULASYONU

Şefik Emre Eskimez

Mekatronik Mühendisliği Programı, Master Tezi, 2013

Tez Danışmanı: Assoc. Prof. Dr. Kemalettin ERBATUR

Anahtar Kelimeler: İnsansı Robotlar, nesne manipulasyonu, nesne kavraması, görme sistemleri, yön bulma

ÖZET

İnsansı robotlar insanların günlük yaşamlarına yardımcı olmaları için dizayn edilmiştir. Bu robotların fiziksel iş isteyen mesleklerde insanların yerini alması bekleniyor. Ayrıca robotların sağlık hizmetleri sektöründe de rol alması düşünülüyor. İnsansı robot bilimindeki nihai amaç robotların insanlarla tam olarak iletişim kurabilmeleri ve toplumun iş gücünde yer almalarıdır.

Sıradan bir insanın günlük hayatını geçirdiği ortam bir çok farklı renkte ve dokuda nesneleri içerir. Bu tip nesneler robot tarafından gerektiğinde kolayce tespit edilebilmeli, nerede olduğu kestirilebilinmeli ve müdahale edilinebilmeli. Bu işleri yapabilmek için ortamdan yüksek miktarda veri alınmalı.

Bilgisayarla görme alanı görsel özelliklerin resimden çıkarılması ve bu verinin bilgisayar sistemleri için kullanılması konusunda çalışmaktadır.

Kameralarla alınan veriler diğer sensorlere gore ortamla ilgili karmaşık görevler için gereken verinin büyük bir kısmını taşımaktadır. Bu tip karmaşık görevlerin sağlanabilmsi için hedef nesnenin alınan veride bulunup diğer nesnelerden ayrılması ve nesnenin 3 boyutlu uzaydaki yerinin belirlenmesi gerekir. Bu tip görevlerde objenin robotun görüş alanında kalması gerekmektedir. Bu da görsel geribeslemeyle sağlanabilir.

Bu tez bir humanoid robot için nesne manipulasyonu algorithması sunmaktadır. Hedef nesne olarak üstünde kırmızı bir kare işaretleyici olan beyaz bir nesne kullnılmıştır. Kırmızı karenin köşeleri bulunmuş ve bu köşeler objenin pozunun bulunmasında ve objenin kamera görüş hizasında tutulmasında kullanılmıştır. Elde edilen poz bilgisi navigasyon ve manipulasyon için kullanılmıştır. Bu algoritma insansı robot deney platform SURALP (**S**abancı **Ü**niversitesi **R**obot **A**raştırmaları **L**aboratuvar **P**latformu) üstünde uygulanmıştır.

To my beloved family

ACKNOWLEDGEMENTS

I would like to thank my thesis advisor Kemalettin ERBATUR, for supporting me through my graduate and undergraduate education. He always supports me and believes in me.

I would like to state my appreciation and thank to my thesis committee; Assoc. Prof. Dr. Volkan PATOĞLU, Assoc. Prof. Dr. Albert LEVI, Assoc. Prof. Dr. Meriç ÖZCAN and Assist. Prof. Dr. Hakan ERDOĞAN for their constructive criticism and overall interest that they have shown for my work.

My student colleagues Kaan Can Fidan, Tunç Akbaş, Utku Seven, Ömer Kemal Adak, Selim Özel, Hazar ilhan and Mert Doğar deserve particular thanks for their invaluable support and friendship.

Last but not least, I would like to thank my family, Suat Eskimez and Şebnem Eskimez for their unrequited love and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

1. INTRODUCTION

The research interest in humanoid robots is increasing all around the world as hardware components are getting cheaper and more powerful. Humanoid robots are designed to assist people in their daily life. The ultimate goal of human like robot is to be able to communicate with humans, and to take a role in the division of labor. They can fulfill manual tasks, which require physical manipulations. Dangerous environments for humans can be harmless to humanoid robots. They can assist people who need daily healthcare. They can be used in service industries and also they can be used as private service robots.

Any robot that is considered to be able to assist people in daily life must have the following features: Communication and interaction with living organisms, absolute safety for humans and environment, to be able to achieve high level tasks. Communication and interaction can be achieved with any robot with little modification such as a camera, microphone and speakers. But to be able to truly communicate with humans, robot must have the body language such as gestures, facial expressions and any part of body movements. The environment that we are living in mostly created by humans, which means the tools, equipments, vehicles etc. is designed for human use. To interact with our environment and to be able to communicate with us, robots require human body form. This is the main motivation behind humanoid robot research.

Achieving high level tasks is a challenging task with humanoid robots. Having too many Degrees of Freedom (DoF) causes the humanoid robot dynamics to be nonlinear. Usually high level tasks are completely automatic, meaning that the robot should perform those tasks unsupervised. Those tasks require humanoid robot to be able to make decisions and overcome the environmental disturbances. Robot should have reliable sensors to be able to achieve those tasks to gather required environmental information.

Vision sensors can retrieve images containing high amount of visual cues about the environment. Feature detectors can be used to extract the information on a task specific environment or object. Human environment is designed mostly favoring the sight (vision) sense;

therefore high level tasks can be achieved by using cameras. But the extracted data highly depends on the hardware quality of cameras and computational costs of computer vision algorithms.

Computer vision is a broad research area that currently has a high amount of research interest worldwide. The researchers are trying to overcome problems that come with hardware power insufficiency and they are trying to minimize the computational costs of vision algorithms as well as finding new solutions to the computer vision problems. Some of the problems in computer vision are as follows: 3D Pose Estimation, Segmentation, Feature Extraction, Object Recognition, Object Manipulation, Face Recognition, Gesture Recognition, Self Localization and Navigation.

Pose estimation is a problem of finding the target object's position and orientation in 3D environment with respect to the camera frame. Object can then be manipulated by the robot with known translation and rotation vectors. In order to be able to estimate the pose, object must be detected and features of the object should be extracted. Segmentation is the problem of labeling the pixels of the image according to the similarities between them, resulting with a set of segments which represents the image in a more meaningful way. A segmented object can be used for feature extraction algorithm. Resulting features can be used for the pose estimation algorithm. Feature extraction reduces the data into relevant and small set of data that can be used for computer vision algorithms. Some example features that can be extracted are corners, edges, blobs, lines, circles on the image.

Vision based robot control (also called visual servoing) is the control approach that uses visual data obtained from cameras to drive the servo motors of the robot. Gaze control can be described as a specialized visual servoing, which allows the humanoid robots head to track objects (Cameras assumed to be mounted on the head of the robot). Gaze control allows humanoid robots to keep the object or other visual targets in the image frame, so that robot can continue to operate on them.

In this thesis, an object manipulation algorithm using single camera is developed and implemented on humanoid robot SURALP. The algorithm includes segmenting the colored object, finding its corners, creating 3D model of the object, solving correspondence between 3D object points and 2D image points. The pose of the object is found and used for humanoid robot navigation and manipulation of an object. To keep the object in the image frame, gaze control for

SURALP is derived. Gaze control is actively used in the phases of navigation and object manipulation. The experimental results are presented for both successful and unsuccessful attempts.

The thesis is organized as follows.

The next chapter presents a survey of the object manipulation in robotics as well as of famous humanoid robots.

Chapter 3 briefly explains the full-body humanoid robot SURALP in terms of hardware components, walking reference generation for both straight walk and circular arc shaped walk, and control architecture. SURALP is the experimental platform that is used in this thesis for implementation.

Chapter 4 explains the computer vision algorithms that are used in this thesis. HSV color space is explained with application to color object segmentation. Morphological operations are described. Harris corner detector is explained in detail. Pinhole camera model and perspective projection terminologies are introduced. Camera calibration process is explained in terms of pinhole camera model. Distortion models are explained and distortion coefficients are described. Last section of this chapter explains the pose estimation algorithm in detail.

In Chapter 5, visual servoing terminologies are introduced. Image based visual servoing (IBVS) control scheme is explained in detail. This chapter briefs gaze control history and applications. From IBVS, gaze control is derived in mathematical terms. Both simulation and experimental results for gaze control are presented.

Chapter 6 explains the overall algorithm for object manipulation and presents experimental results.

Chapter 7 discusses experimental results, future work and improvements to this study.

The last chapter contains a summary of the work done in the thesis.

Chapter 2

2. LITERATURE REVIEW ON OBJECT MANIPULATION

This chapter contains a literature review on object manipulation in robotics. Also some humanoid robots which accomplish high level tasks using visual information are introduced. Object manipulation is analyzed for three different kinds of robots; namely stationary robots, wheeled robots and humanoid robots.

Object manipulation is mandatory for any robot to operate in our daily life. Most of the object manipulation systems depend on visual sensors as these sensors retrieve the most of the information about the 3D environment.

2.1. Stationary Robots

Stationary robots are fixed in position. Industrial robot arms such as PUMA 560 are falls in this category. [1] achieved object tracking with PUMA 560 end manipulator with visual feedback. IBVS control scheme was used to track the object with a camera that is mounted on the manipulator arm. Manipulator arm could track moving objects successfully. On a later work by [2], affine stereo vision algorithm is used to locate and reach the object with a 5 DoF robot arm. [3] proposed close loop visual feedback for grasping the object. Object grasping is achieved by using an independent camera directed to the manipulator arm (eye-to-hand configuration) with visual servoing. [5] proposed a semi- automatic robotic grasping system that merges visual feedback and online grasp planning.

Assistance robots that help people in object manipulation tasks must be mentioned although they do not use visual feedback. They are a stepping stone for the ultimate goal. Some of the assistance robots are Master [6], MySpoon [7], HANDY 1 [4] and PROVAR [5]. They relied on the sensor feedback to grasp the object. However, because they are fixed to a position, their workspace is highly limited. More information on rehabilitation robots can be found in [42].

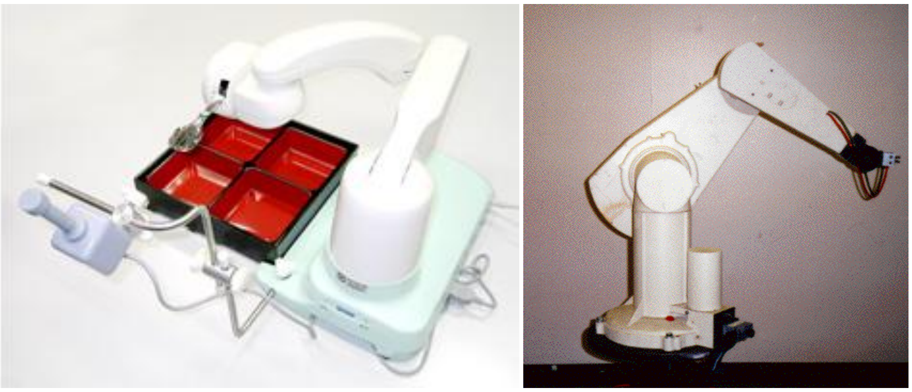


Figure 2.1 On the left MySpoon assistance robot, on the right puma 560 robot arm ([6] and [1])

2.2. Mobile robots

Some of the important works that include object grasping with wheeled robots are reported in [10-12]. These wheeled platforms are equipped with robotic arms. They rely on their visual sensors for navigation and object manipulation. [10] introduces a mobile robot that can open doors and navigate through them using visual information to find the door knob. [11] has a similar robot that can navigate in buildings and can use the elevator. [12] uses visual servoing to grasp objects with a manipulator mounted on mobile robot.

Most important mobile assistance robot systems are Tou [13], ISAC [14], Manus, Raptor [15], FRIEND II [16] and the VICTORIA project [17]. These robots mostly depend on visual information for object grasping. Some of them are using multi cameras in controlled environments.

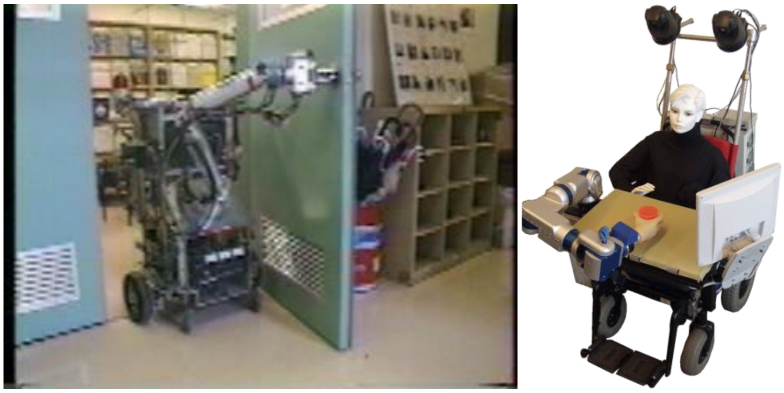


Figure 2.2 On the left YAMABICHO robot in [10], on the left FRIEND II assistance mobile robot [16].

2.3. Humanoid Robots

In this section, firstly, humanoid robots that achieve high level tasks are introduced. Next, the works on object manipulation using humanoid robots are presented.

2.3.1. Important Humanoid Robots

The most famous humanoid robot is ASIMO (Advance Step in Innovative Mobility) [18]. It is a part of HONDA motor company's humanoid robot family. ASIMO has 26 DoF, its weight and height is 52 kg and 1.2 m respectively. ASIMO has a technology called i-WALK which enables it to walk while interacting with the environment [19]. Some of the ASIOMO's capabilities are distinguishing sounds, facial recognition, environmental recognition, recognition of posture and gesture, recognition of moving objects, object manipulation. ASIMO can identify 10 different faces and can respond to each of them by their names. It can greet a person then follow him or her. ASIMO can detect a bottle and pour the contents of the bottle to a glass and serve it. Figure 2.3 shows the humanoid robot ASIMO. Figure 2.4 shows the other member of the family of humanoid robots of HONDA motor company.

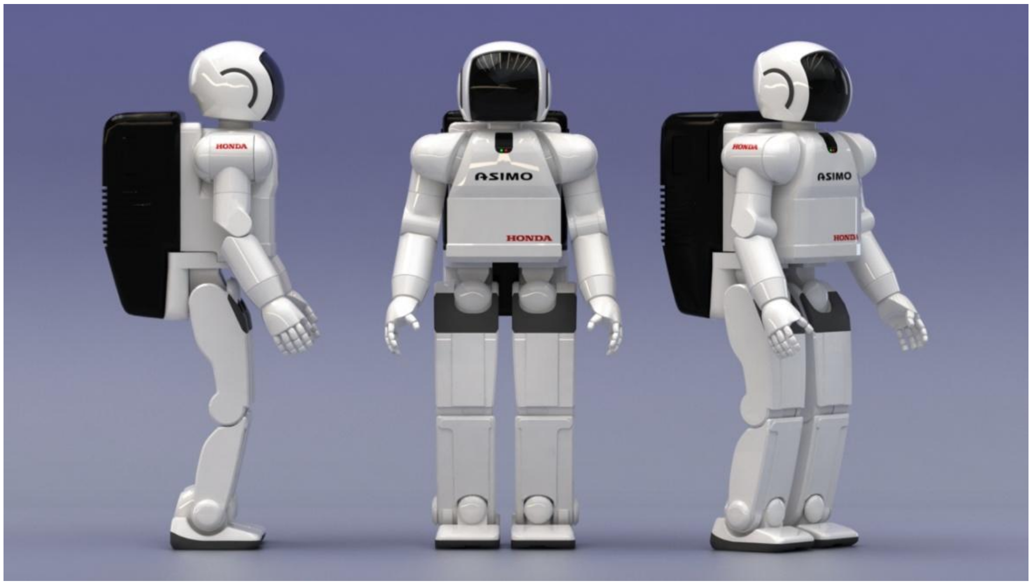


Figure 2.3 HONDA's humanoid robot ASIMO [18]

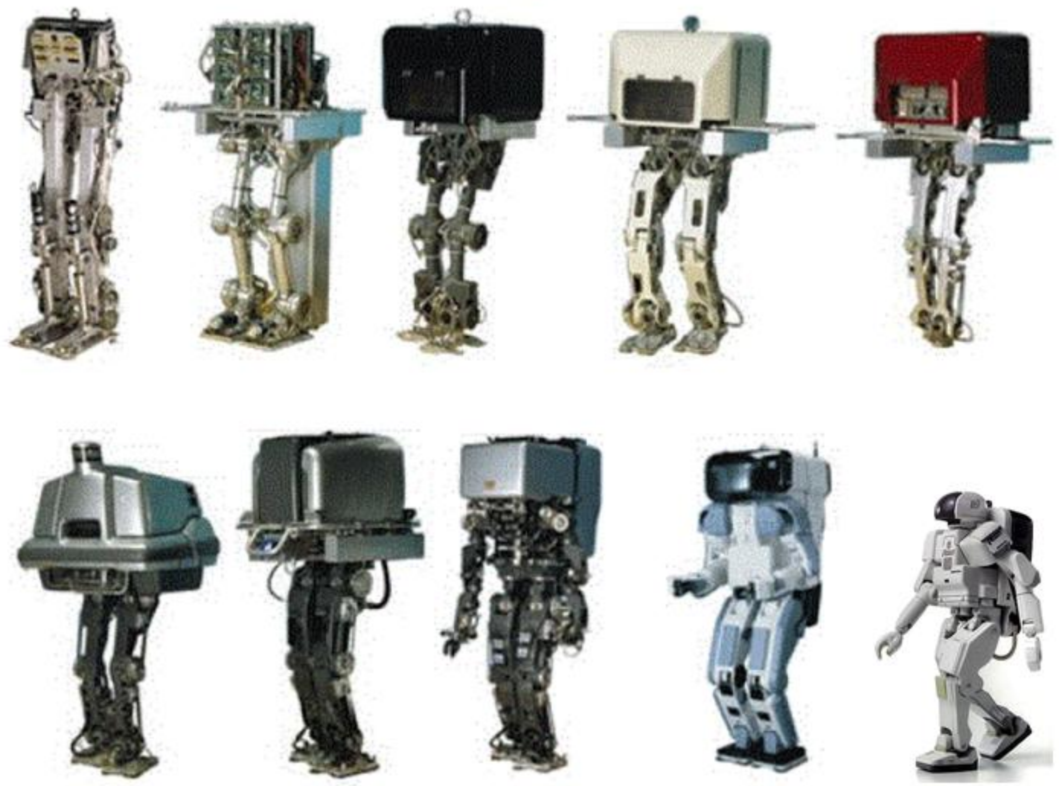


Figure 2.4 HONDA motor company’s humanoid robot family [79]

University of Tokyo has humanoid robot prototypes, namely H5, H6 and H7. H5 was a 30 DoF child size humanoid robot. It was incapable of full body motions. H6 was designed to be capable of environmental interaction [20]. It had 3D vision capabilities. Its 35 DoF enabled the robot to perform full body motion tasks. H7 is the current prototype. It has stereo vision capabilities enabling arm motion planning for high level tasks [21]. Figure 2.5 shows the robot family H5-7.

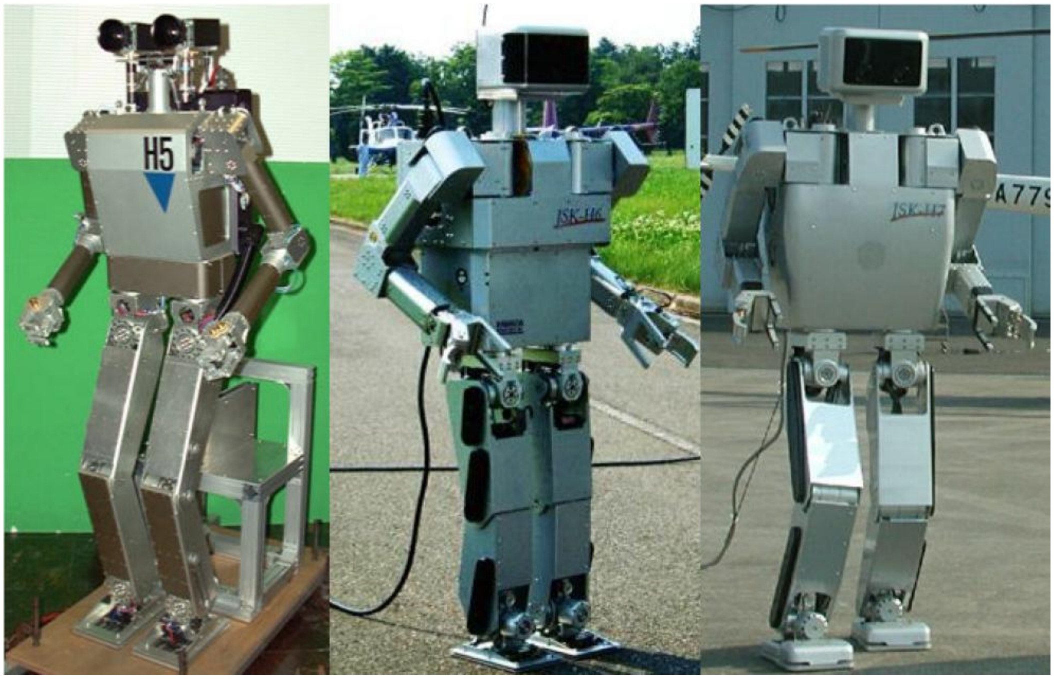


Figure 2.5 Tokyo University’s humanoid robot prototypes: H5-7 (from left to right) [20-21]

In 1998, the Ministry of Economy and Industry (METI) of Japan announced the Humanoid Robot Project (HRP) with the support of New Energy and Industrial Technology Development Organization (NEDO), Kawada Industries and National Institute of Advanced Industrial Science and Technology (AIST). The first robots are P3 models which are bought from HONDA motor company [22]. HRP humanoid robot series is shown in Figure 2.6 [23]. The feature that came with HRP-2 was the lack of a backpack on the humanoid robot's body. It has a thinner body unlike the previous humanoid robots. HRP-2 can generate a grid-based map of the environment with a stereo vision system that is mounted on its head [24]. HRP-4 is the current prototype. One of the main features of the HRP series robots is that they can lift from the ground after lying down.

Another important humanoid robot is KHR (KAIST Humanoid Robot) series. KHR-1 is their first prototype and it has 25 DoF [25]. It is followed by KHR-2, which was able to walk on uneven surfaces and inclined floor [26]. KHR-3, also known as HUBO, is their current prototype [27]. It is commercially available and can be used for achieving high level tasks. Figure 2.7 shows the KHR family.

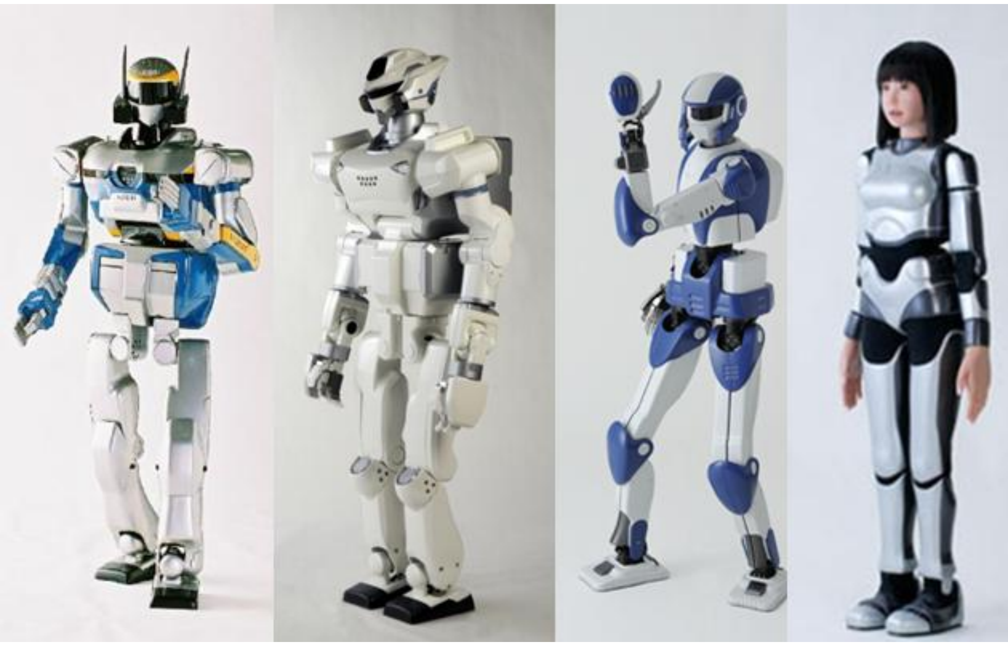


Figure 2.6 -From left to right: HRP-2, HRP-3, HRP-4 AND HRP-4C [22-24]

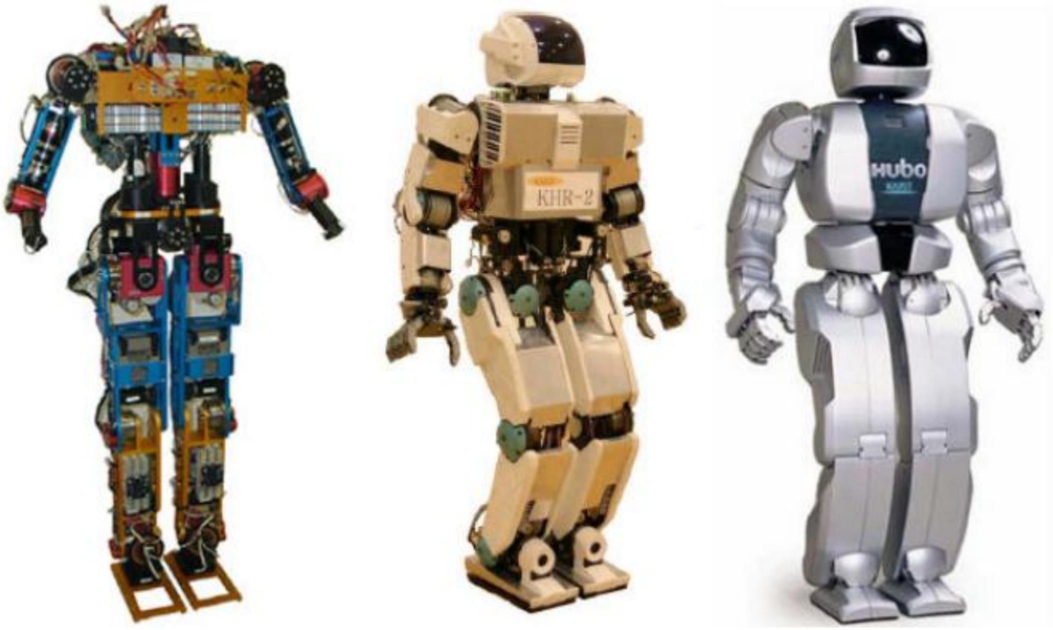


Figure 2.7 From left to right: KHR-1, KHR-2 and KHR-3 (also known as HUBO) [25-27]

High level tasks such as stair climbing and obstacle avoidance with humanoid robots are reported in [27-30]. Humanoid robot kicking a ball from stationary position is reported in [31]. Also frameworks which enable humanoid robots to do high level tasks are reported in [32-33].

2.3.2. Object Manipulation by Humanoid Robots

Early work on humanoid robots toward object manipulation is reported in [35]. Authors used a 35 DoF humanoid robot with 4-DoF arms. Figure 2.8 shows this humanoid robot. This robot could grasp an object even though the object was not on the workspace of its current position. It reached the object with changing upper body posture. The vision system they used to find the object was based on the image block matching algorithm.

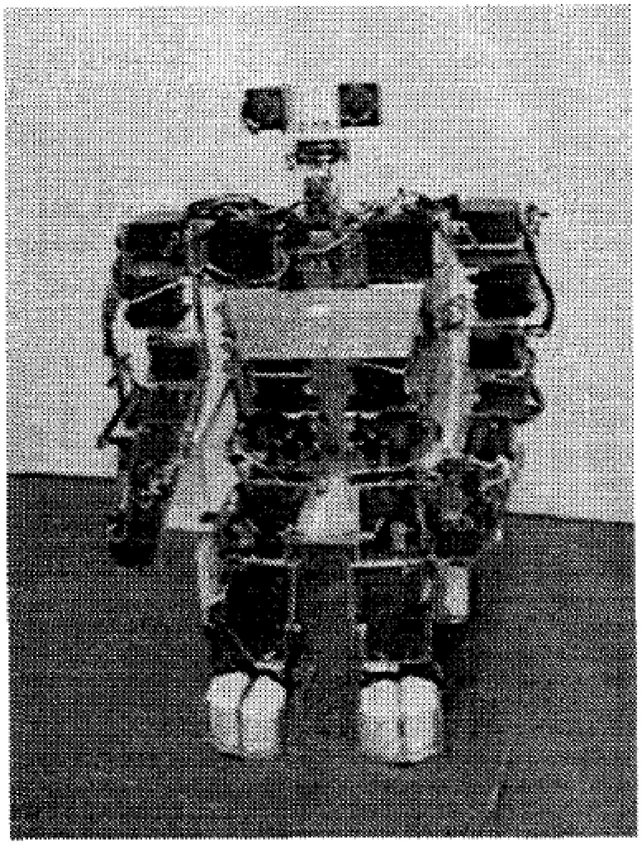


Figure 2.8 35 DoF humanoid robot [35]

[36] used a humanoid robot torso (Magilla) for grasping objects. They used a closed-loop haptic control model with visual feedback. The experiments mainly focused on haptic and visual learning.

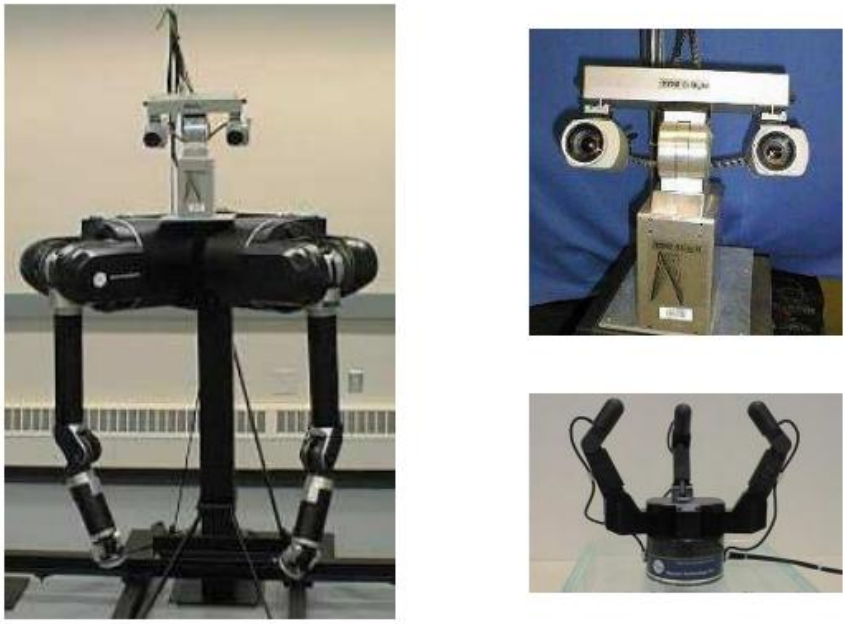


Figure 2.9 Magilla, the UMASS humanoid torso [36]

[37] reported object manipulation with visual servoing by a humanoid robot. Authors proposed a model-based gripper pose estimation using Kalman filter. They used position-based visual servoing with solutions to classic drawbacks that came with it. Active vision was used to track the objects. Stereo cameras were used. The humanoid robot platform that the authors used is shown in Figure 2.10. Few years later, the authors proposed a hybrid visual servoing control scheme to improve the accuracy of the manipulation tasks [38]. The system had robustness to occlusions that was caused during manipulation.

A virtual humanoid robot platform that can achieve navigation and obstacle avoidance was presented in [39]. The virtual humanoid robot achieved these tasks by using visual servoing. It could find the pose between objects and tracked them. Although object manipulation was not implemented at the time, the framework that they created could be used for object manipulation. Figure 2.11 illustrates the virtual humanoid while tracking an object.

Another object grasping achieved by a virtual humanoid robot was reported in [40]. H6 humanoid robot's virtual model was used to plan collision free path to grasp an object. A virtual vision system was used to determine if the object was in range or not. The authors developed Rapidly-exploring Random Trees (RRTs) for manipulation planning. The resulting system was also implemented successfully on the H6 humanoid robot.

A system that could achieve full body reaching motion by humanoid robots was described in [41]. The visual system extracts the target information by color information. A yellow toy bat with a light blue grip was used as a target object. 3D pose information of the object is extracted using a stereo vision system. The robot could execute various reaching motions as shown in Figure 2.12.

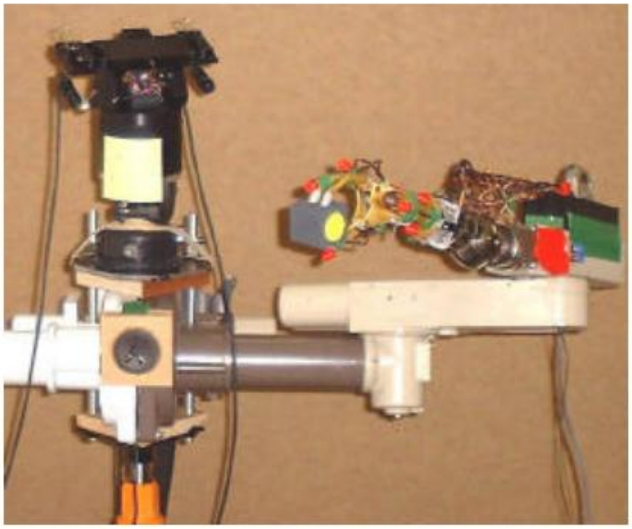


Figure 2.10 Humanoid robot platform that was used for visual servoing experiments in [37]

An object grasping while walking task was implemented with a humanoid robot in [43]. The authors used a structure called stack of tasks which enables easy access for task sequencing. This structure was used for task sequencing, including walking, grasping and visual servoing tasks. For experiments, they used HRP-2 robot.

[44] worked on neural networks to teach reaching motion to a humanoid robot. The goal was object manipulation when the robot learned to reach. They used visual servoing for generating reaching motion. They also used gaze control to fixate the object on the corner of the image. The humanoid robot James was used in the experiments. The model did not rely on prior information about the kinematic structure of the robot. The robot's hand was located visually by using color markers.

Grasping of complex textured objects using stereo cameras by a humanoid robot was discussed in [45]. Humanoid robot ARMAR was used in experiments. Authors employed object recognition that uses object shape for recognition. This research enabled working with realistic objects in the object manipulation field. In another work of the authors, a hybrid approach, which combines visually retrieved and kinematically determined pose of the manipulator, was used to obtain robust object manipulation [46]. Proposed method was implemented on the humanoid robot ARMAR III. Another work that used the ARMAR humanoid robot's head and a robotic manipulator recognized, located and grasped the object [47]. Authors used four cameras, two for a wide field set for finding the object and a foveal set for recognition and manipulation. The information of the object was gathered before and after the grasping action for learning purposes.

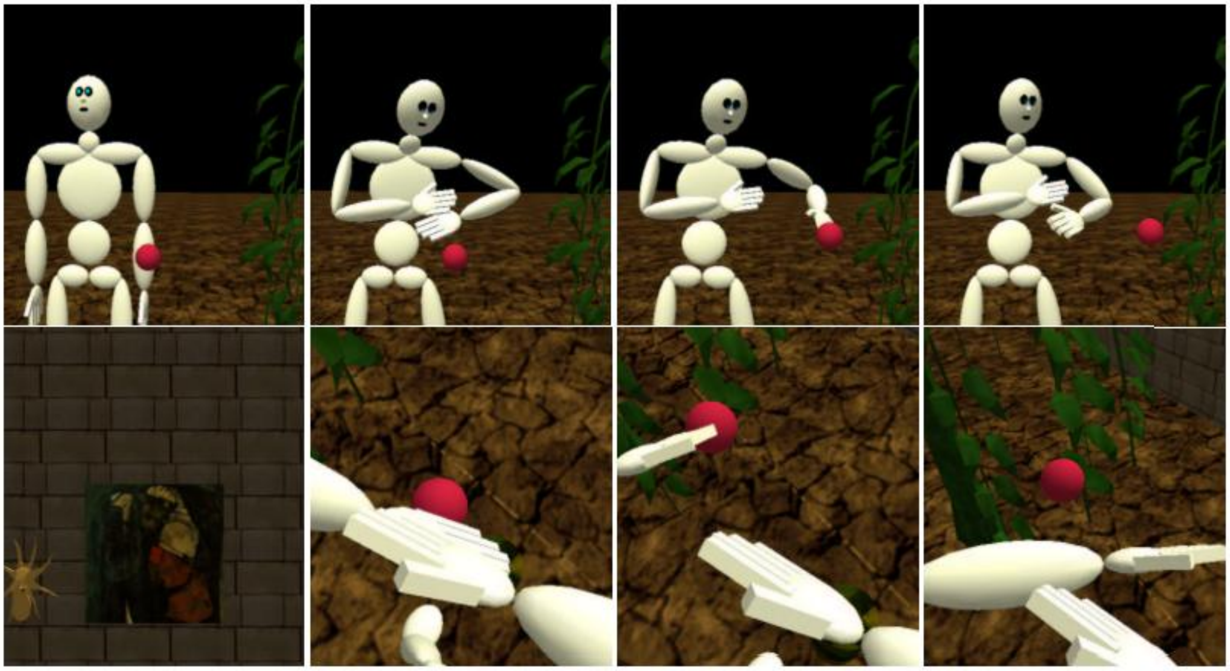


Figure 2.11 Virtual Humanoid robot gazing at a moving object [39]

Combination of visual feedforward and feedback system was reported in [48]. The system was implemented on humanoid robot BHR-02 for object grasping tasks. In experiments they used red ball and a marker fixated on the robot's hand. The combined system reduced the manipulation time with respect to conventional methods.

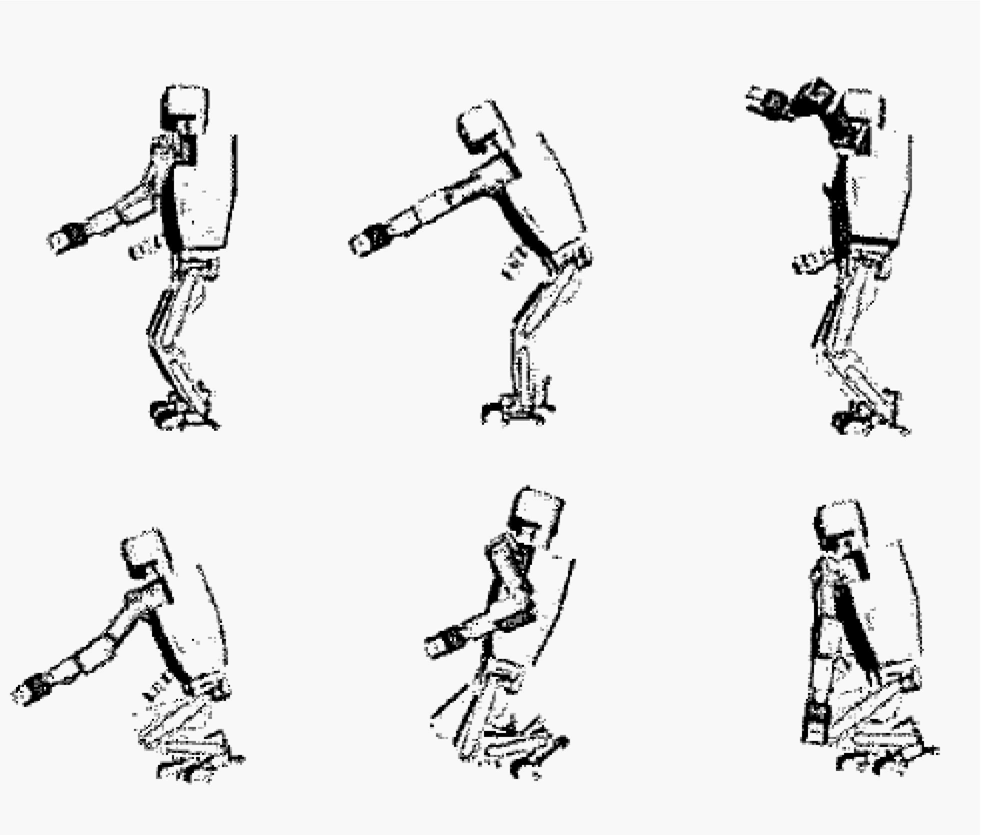


Figure 2.12 Reaching positions implemented by [41]

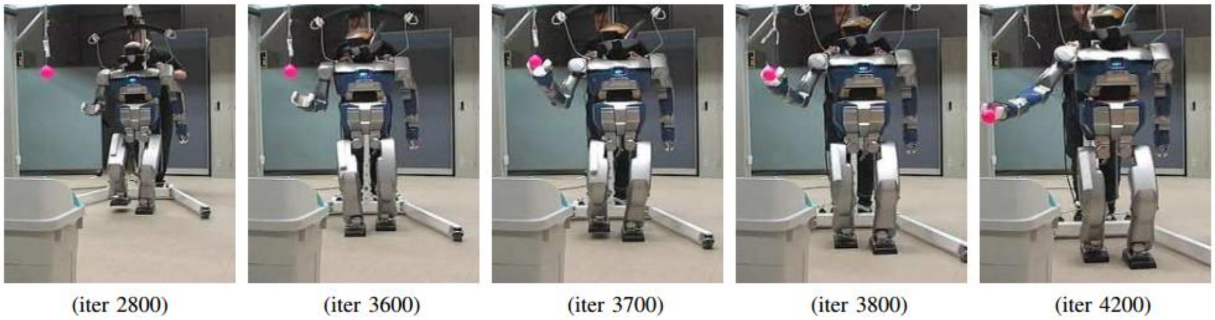


Figure 2.13 HRP-2 is grasping object while walking [43]

Chapter 3

3. EXPERIMENTAL SETUP: SURALP, A FULL BODY HUMANOID ROBOT

SURALP is a full body humanoid robot that is designed for biped locomotion experiments in Sabanci University. In this chapter SURALP is introduced in terms of hardware, walking reference generation pattern, circular arc shaped path reference generation pattern and control architecture.

3.1. Hardware

In this section SURALP's hardware properties are described in terms of mechanical design, sensors, controller hardware and actuation mechanism. A picture of SURALP has been shown in Figure 3.1.

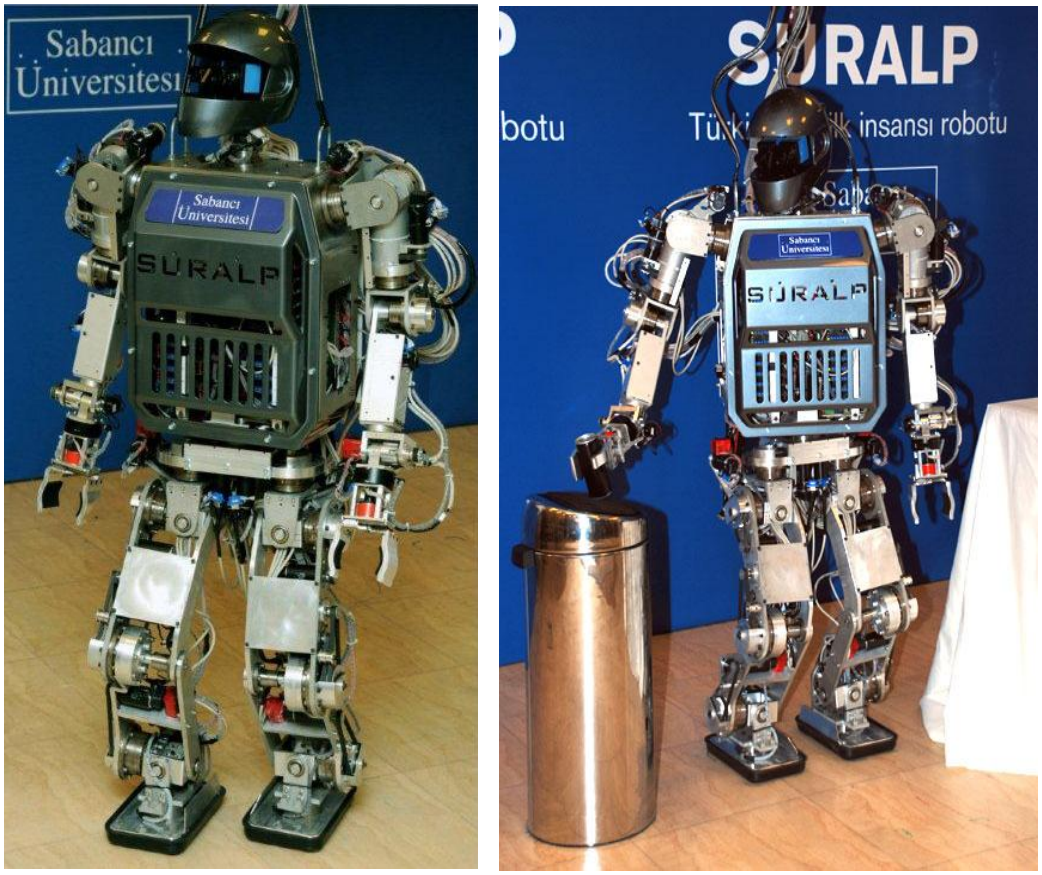


Figure 3.1 SURALP: a full body humanoid robot [49]

SURALP has 29 degrees of freedom (DoF), including 6-DoF legs, 6-DoF arms, 2-DoF neck, 1 DoF hands and 1-DoF waist [49]. Robots weight is 114 kg and its height is 1.66 m. Dimensions of all links are shown in Figure. 3.2 and link lengths are shown in Table 3.1. Its controller hardware (motor drivers) is placed in its trunk. Actuators of SURALP are DC motors. Motors are connected with Harmonic Drive (HD) reduction gear via belt-pulley systems. Table 3.2 shows the motor powers and HD reduction rates. For each joint, only one DC motor is used except knee joint. Knee joint are driven by two DC motors which satisfies the high torque requirement for bipedal gait.

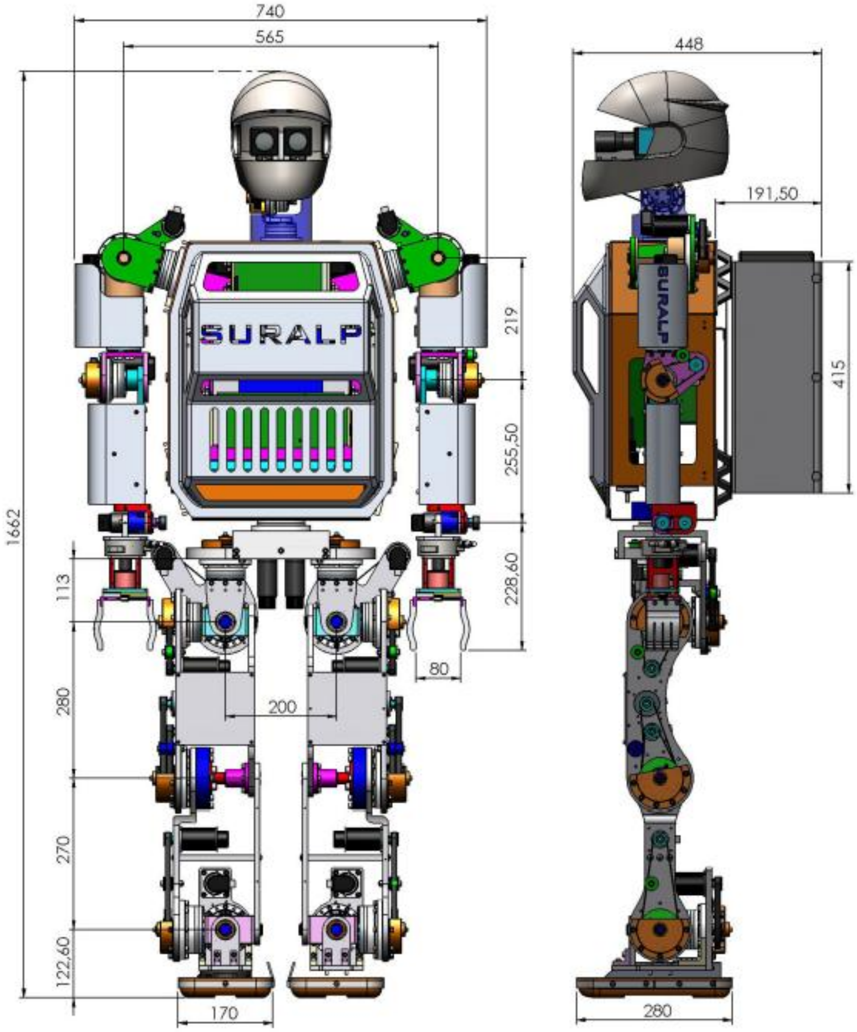


Figure 3.2 SURALP link dimensions [49]

Table 3.1: Length of links

Upper Leg Length	280mm
Lower Leg Length	270mm
Sole-Ankle Distance	124mm
Foot Dimensions	240mm x 150mm
Upper Arm Length	219mm
Lower Arm Length	255mm

Table 3.2: Joint actuator specifications

Joint	**Motor Power**	**Pulley Ratio**	**HD Ratio**	**Motor Range**
Hip-Yaw	90W	3	120	-50 to 90 deg
Hip-Roll	150W	3	160	-31 to 23 deg
Hip-Pitch	150W	3	120	-128 to 43 deg
Knee 1-2	150W	3	160	-97 to 135 deg
Ankle-Pitch	150W	3	100	-115 to 23 deg
Ankle Roll	150W	3	120	-19 to 31 deg
Shoulder Roll 1	150W	2	160	-180 to 180 deg
Shoulder Pitch	150W	2	160	-23 to 135 deg
Shoulder Roll 2	90W	2	120	-180 to 180 deg
Elbow	150W	2	120	-49 to 110 deg
Wrist Roll	70W	1	74	-180 to 180 deg
Wrist Pitch	90W	1	100	-16 to 90 deg
Gripper	4W	1	689	0 to 80 mm
Neck Pan	90W	1	100	-180 to 180 deg
Neck Tilt	70W	2	100	-24 to 30 deg
Waist	150W	2	160	-40 to 40 deg

The controller hardware of robot is constructed with dSPACE modular hardware. Central controller of SURALP's controller mechanism is DS1005 board. Control loop is processed on this board. To be able to retrieve and process the joint encoders, DS3001 incremental encoder input boards are used. A/D and D/A conversions of sensors are done by DS2002 board and DS2103 board respectively.

The sensor system of SURALP is shown on Table 3.3. Each DC motor has a incremental optic encoders to measure the motor angular position. Robots ankles and wrists contain 6 axis force/torque sensors. Inside of robots trunk, a gyroscope, a linear accelerometer and an inclinometer are placed and they form the inertial measurement system of robot. Two Basler 601fc Firewire cameras are mounted to the head of SURALP and communication is made with control desktop via Firewire cables.

Table 3.3 Sensory system of SURALP

	Sensor	**Number of Channels**	**Range**
All joints	Incremental optic encoders	1 channel per joint	500 pulses/rev
Ankle	F/T sensor	6 channels per ankle	± 660 N (x, y-axes) ± 1980 N (z-axis) ± 60 Nm (all axes)
Torso	Accelerometer	3 channels	± 2 G
	Inclinometer	2 channels	± 30 deg
	Rate gyro	3 channels	± 150 deg/s
Wrist	F/T sensor	6 channels per wrist	± 65 N (x, y-axes) ± 200 N (z-axis) ± 5 Nm (all axes)
Head	CCD camera	2 with motorized zoom	640x480 pixels (30 fps)

Kinematic arrangement of SURALP is shown on Figure. 3.3. The Denavit-Hartenberg table for 6-DoF leg is shown in Table 3.4. Axis assignments of Table 3.4 are shown in Figure 3.4.

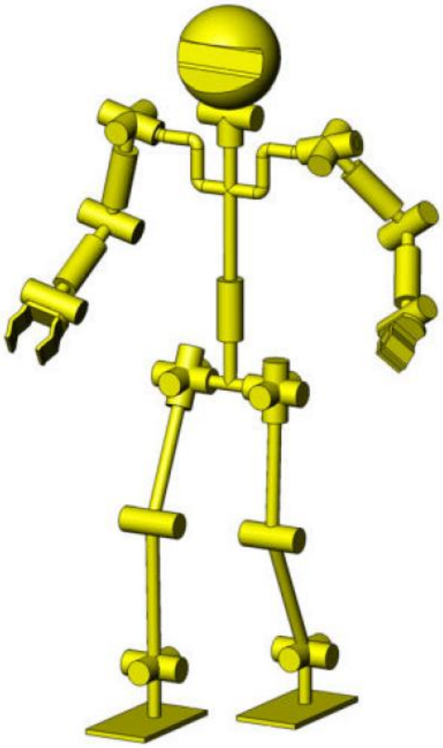
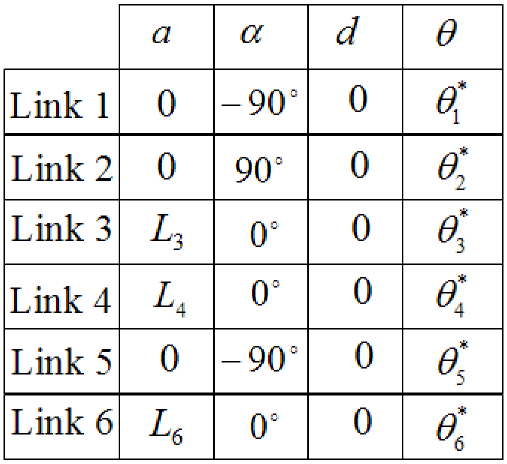


Figure 3.3: Kinematic arrangement of SURALP [49]

Table 3.4: Denavit Hartenberg table for 6-DoF leg



	a	α	d	θ
Link 1	0	-90°	0	θ_1^*
Link 2	0	90°	0	θ_2^*
Link 3	L_3	0°	0	θ_3^*
Link 4	L_4	0°	0	θ_4^*
Link 5	0	-90°	0	θ_5^*
Link 6	L_6	0°	0	θ_6^*

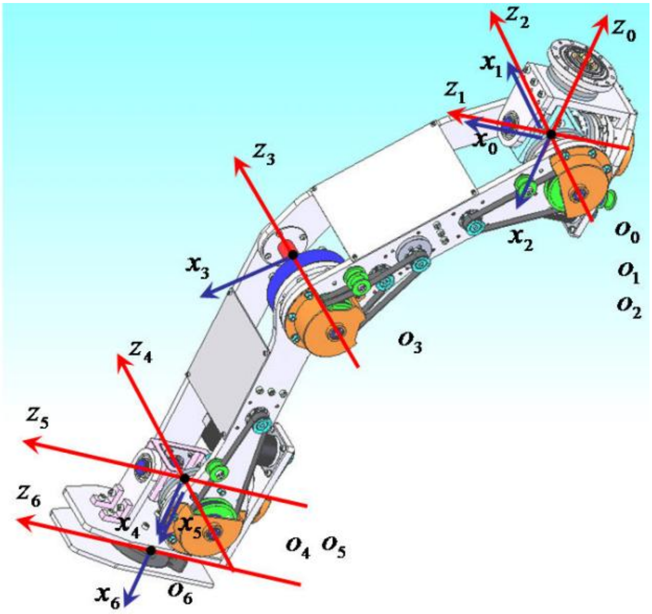


Figure 3.4: Denavit-Hartenberg axis assignment for 6-DoF leg

The hardware hierarchy is presented in Figure 3.5.

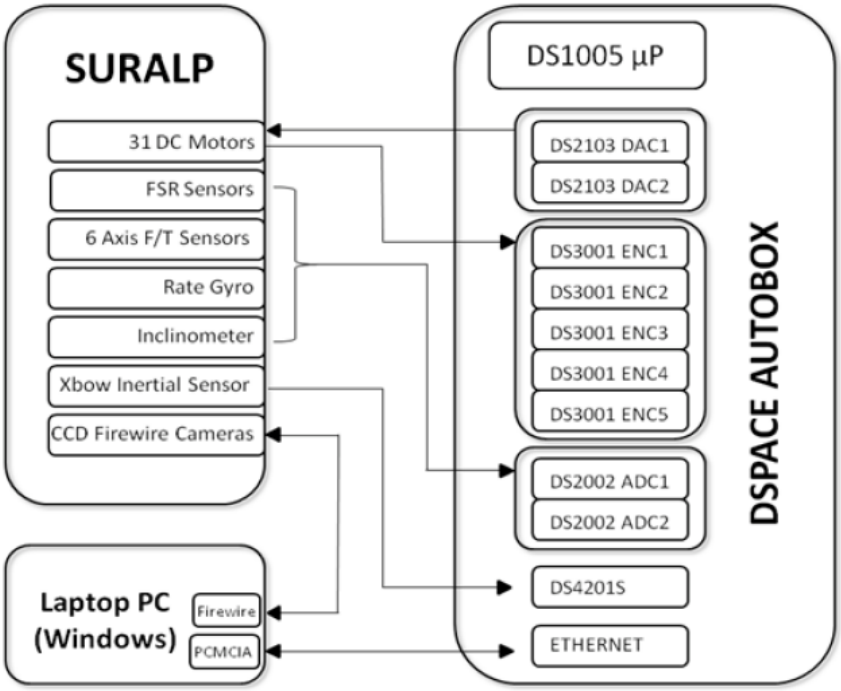


Figure 3.5: The complete hardware architecture of SURALP [49]

3.2. Walking Reference Generation Based on Zero Moment Point

In this section, reference generation for straight walking pattern and walking on circular arc shaped path pattern are described. To generate such references, Linear Inverted Pendulum Model (LIPM) is used [50].

LIPM models the robot with a single mass placed on the robots Center of Mass (CoM) that is linked to a stable contact point on the ground with a massless rod. Figure 3.6 shows LIPM, where $c = (c_x, c_y, c_z,)^T$ is the position vector of the point mass. The massless rod is representing the supporting leg during the walk. A linear system is obtained assuming robot CoM is at a fixed height (c_z is constant). Also swing leg is assumed to be massless during the walk. Figure 3.7 shows the biped model that is used for reference generation.

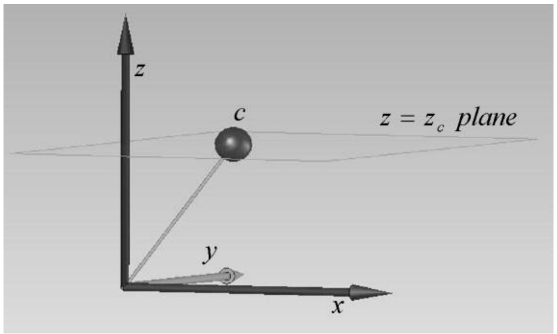


Figure 3.6: The linear inverted pendulum model (LIPM) [50]

The ZMP is defined as a point on the *x-y* plane where horizontal torque components do not exist. Using the structure from Figure 3.6, the expressions for ZMP coordinates p_x and p_y are [ZMP],

$$p_x = c_x - \frac{z_c}{g}\ddot{c}_x \tag{3.1}$$

$$p_y = c_y - \frac{z_c}{g}\ddot{c}_y \tag{3.2}$$

z_c is the height of the plane which the motion of the point mass is constrained. g is the gravity constant: ZMP trajectories can be constructed for reference generation using Equation 3.1 and 3.2 [51-52]. Stability criteria is defined as follows, ZMP should always lie in the supporting polygon defined by the foot (or feet) touching to the ground [53]. In Figure 3.8, an example of fixed ZMP reference trajectory is shown. *A* is the distance between the foot centers in the y direction, *B* is the step size and *2T* is the walking period. [50] and [54-55] shows that ZMP of natural human walk moves forward under the foot sole. Figure 3.9 shows forward moving ZMP reference trajectory with pre-assigned double support phases. Double support phase is defined with the parameter τ.

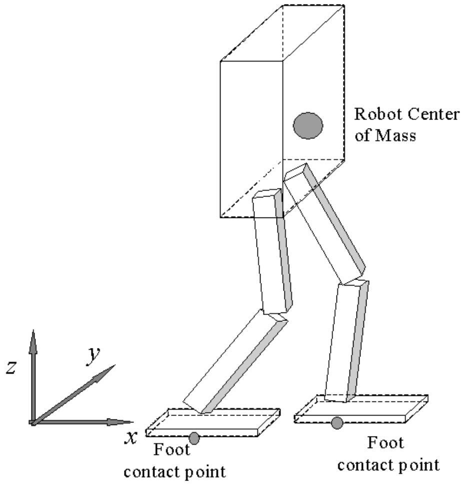


Figure 3.7: Single mass robot model [49]

With defined mathematical functions of ZMP references $p_x^{ref}(t)$ and $p_y^{ref}(t)$, CoM reference functions $c_x^{ref}(t)$ and $c_y^{ref}(t)$ are derived. In this process, Fourier series of the ZMP references are used. The obtained CoM reference function in the x direction is [53],

$$c_x^{ref} = \frac{B}{T}\left(t - \frac{T}{2}\right) + \frac{\alpha_0}{2} + \sum_{n=1}^{\infty} \alpha_k \cos\left(\frac{2\pi nt}{T}\right) + \beta_k \sin\left(\frac{2\pi nt}{T}\right) \tag{3.3}$$

The coefficients α_0 and α_k(k = 1, 2, 3…) are zero. The coefficient β_k is

$$\beta_k = \frac{w_n^2 T^2}{\pi^2 k^2 + w_n^2 T^2} \frac{2}{\pi k} \left\{ \sigma_1 \left[-\tau \cos\left(\frac{2\pi k \tau}{T}\right) + \frac{T}{2\pi k} \sin\left(\frac{2\pi k \tau}{T}\right) \right] + \sigma_2 \left[\tau \cos\left(\frac{2\pi k \tau}{T}\right) - \frac{T}{2} \cos\left(\frac{2\pi k \tau}{T}\right) - \frac{T}{2\pi k} \sin\left(\frac{2\pi k \tau}{T}\right) \right] \right\} \tag{3.4}$$

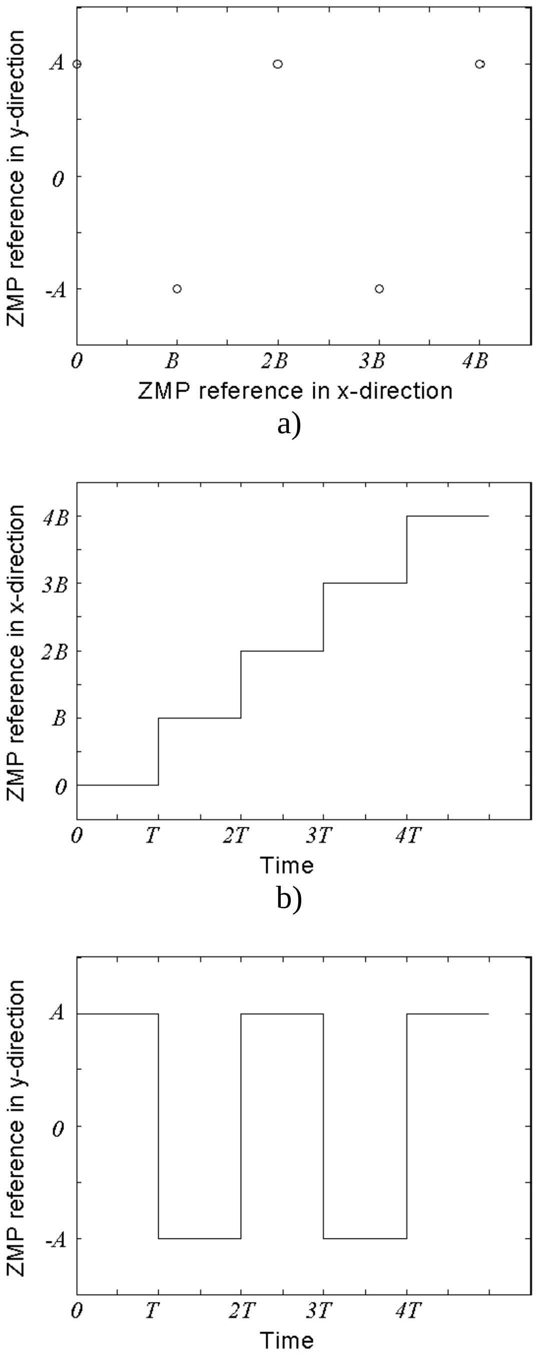
where $w_n \equiv \sqrt{g/z_C}$. Similarly CoM reference function in y direction is [70 again],

$$c_y^{ref} = \frac{\alpha_0}{2} + \sum_{k=1}^{\infty} \alpha_k \cos\left(\frac{2\pi k t}{2T}\right) + \beta_k \sin\left(\frac{2\pi k t}{2T}\right) \tag{3.5}$$

The coefficients α_0 and α_k(k = 1, 2, 3…) are again zero. The coefficient β_k is

$$b_k = \begin{cases} \frac{w_n^2 T^2}{\pi^2 k^2 + w_n^2 T^2} \frac{2A}{\pi k} \left\{ \begin{matrix} \frac{2}{\tau}\left[-\tau \cos\left(\frac{\pi k \tau}{T}\right) + \frac{T}{\pi k} \sin\left(\frac{\pi k \tau}{T}\right) \right] + \\ \left[\cos\left(\frac{\pi k \tau}{T}\right) - \cos\left(\frac{\pi k (T-\tau)}{T}\right) \right] \end{matrix} \right\} & if\ k\ is\ odd \\ 0 \quad if\ k\ is\ even \end{cases} \tag{3.6}$$

The CoM reference functions obtained are shown in Figure 3.10 with corresponding ZMP references.

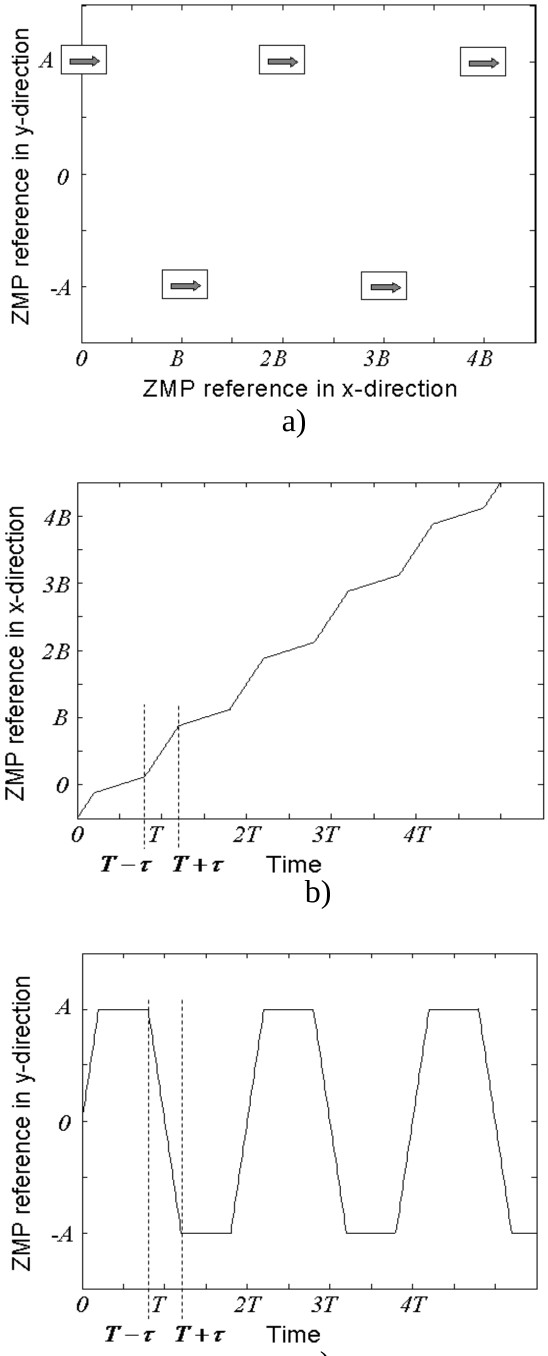


a)

b)

c)

Figure 3.8: Fixed ZMP references. a) $p_x^{ref} - p_y^{ref}$ Relation on the $x - y$ plane b) p_x^{ref} , the x -axis ZMP reference c) p_y^{ref} , the y -axis ZMP reference



a)

b)

c)

Figure 3.9: Forward moving ZMP references with pre-assigned double support phases. a) $p_x^{ref} - p_y^{ref}$ Relation on the $x - y$ plane b) p_x^{ref}, the x-axis ZMP reference c) p_y^{ref}, the y-axis ZMP reference

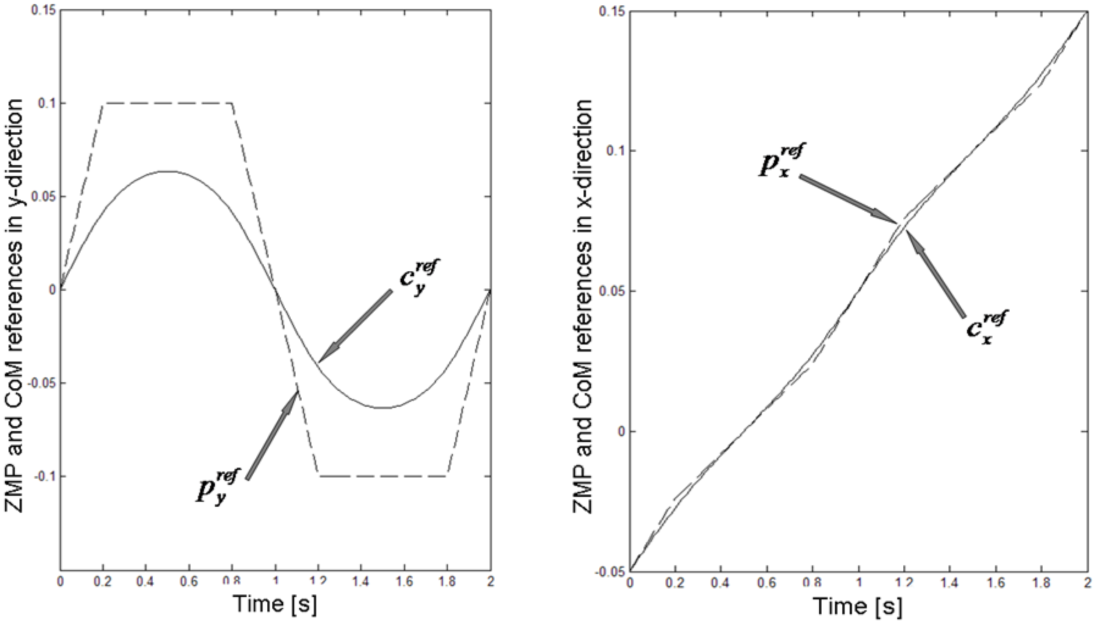


Figure 3.10: x and y -direction CoM references together with the corresponding original ZMP references (A = 0.1 m, B = 0.1 m, b = 0.04 m, T = 1 s and τ=0.2 s)

To be able to use obtained CoM reference functions foot placement must be arranged. This operation includes foot position, foot orientation and foot placement timing. Foot placement components are shown in Figure 3.11 in terms of x and z direction components. T_d and T_s are double and single support periods respectively. h_s is the step height. Trajectories along the y direction are constant and the values are fixed at –A and A for the right and left foot respectively. Foot orientation references arranged to be parallel to the even ground. The reference generation process can be found in [53] and [56] in more detail.

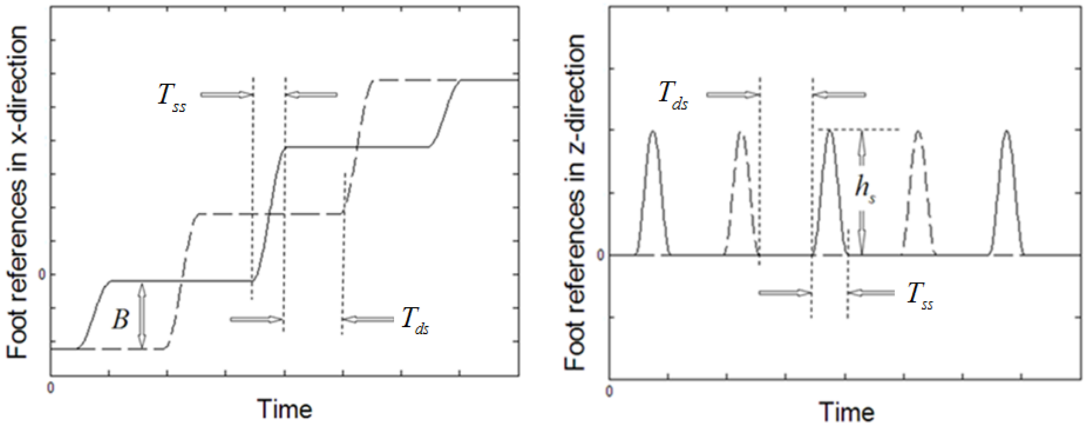


Figure 3.11 Foot frame references expressed in the world frame (x and z direction). Solid curves belong to the right foot and dashed curves belong to the left foot.

3.3. Walking Reference Generation for Circular Arc Shaped Paths

In this section, reference generation that is described for straight walk is mapped to a circular arch path as shown in Figure 3.12.

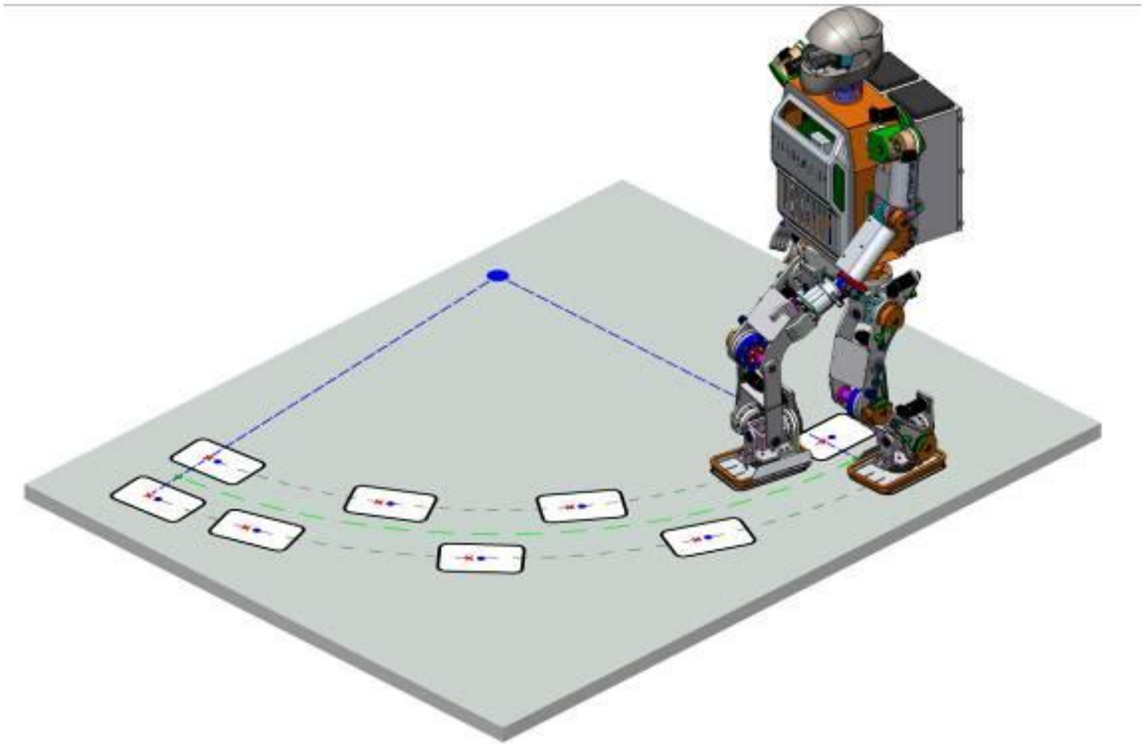


Figure 3.12 SURALP CAD model walking on arc shaped trajectory.

Mapping process starts with assumption that there exist a line which connects projection of initial body frame coordinates on the ground and projection of final body frame coordinates on the ground. Figure 3.13 shows the line and foot placements of straight walk. The distance can be computed as,

$$s = (N-1)B \tag{3.7}$$

The parameter N is the number of swings through the walk. B is defined as step size. This line is mapped to circular arc in Figure 3.14. The angle can be calculated as,

$$\psi_{total} = \frac{(N-1)B}{r_c} \tag{3.8}$$

The parameter r_c is the radius of circle which the turning occurs. At any point on this circle, the body and feet are kept parallel to the arc. If this parameter is very large, the circular walk seems to be a straight walk. In SURALP implementation, positive values of r_c corresponds to right turn and negative values corresponds to left turn.

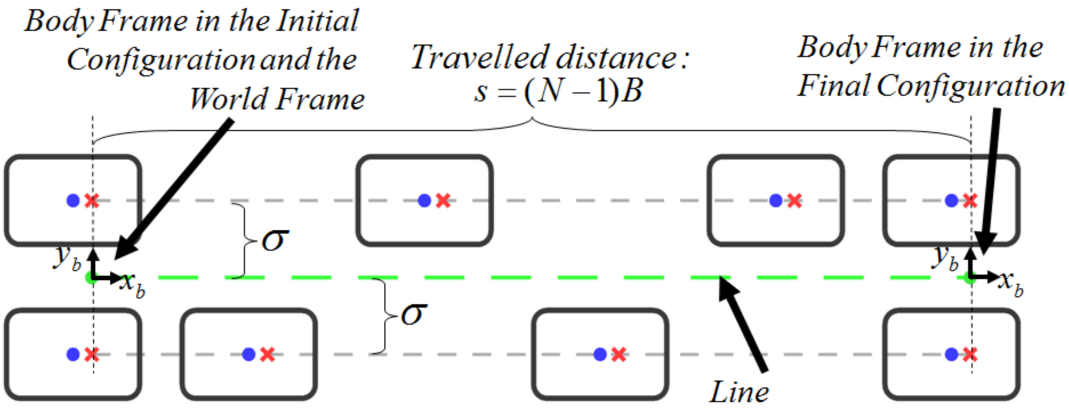

Body Frame in the Initial Configuration and the World Frame
Travelled distance: $s=(N-1)B$
Body Frame in the Final Configuration
σ
σ
y_b
x_b
y_b
x_b
Line

Figure 3.13 Straight walk foot placement locations

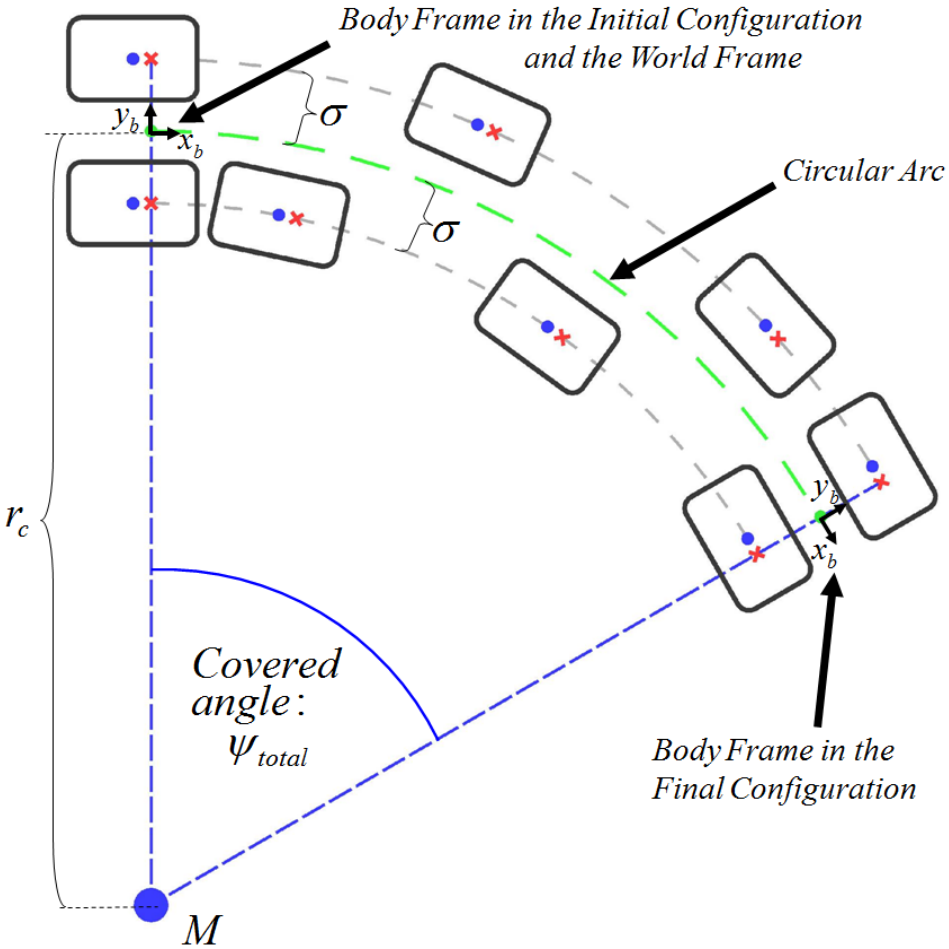

Body Frame in the Initial Configuration and the World Frame
σ
σ
y_b
x_b
Circular Arc
r_c
y_b
x_b
Covered angle: ψ_{total}
Body Frame in the Final Configuration
M

Figure 3.14 Mapping of Straight walk foot placement locations to circular arc path

3.4. Walking Control Algorithm

Walking control scheme is given in Figure 3.15. Firstly, CoM references and foot placement references are generated as described in section (1.2). Then the joint positions are calculated through inverse kinematics. For each joint, separate PID controller is used for joint position control. During the walk phase, a set of sub control routines are activated to compensate the error that is caused from reference trajectories and actual trajectories. These modules are also shown. [49] explains the blocks in Figure 3.15 in detail.

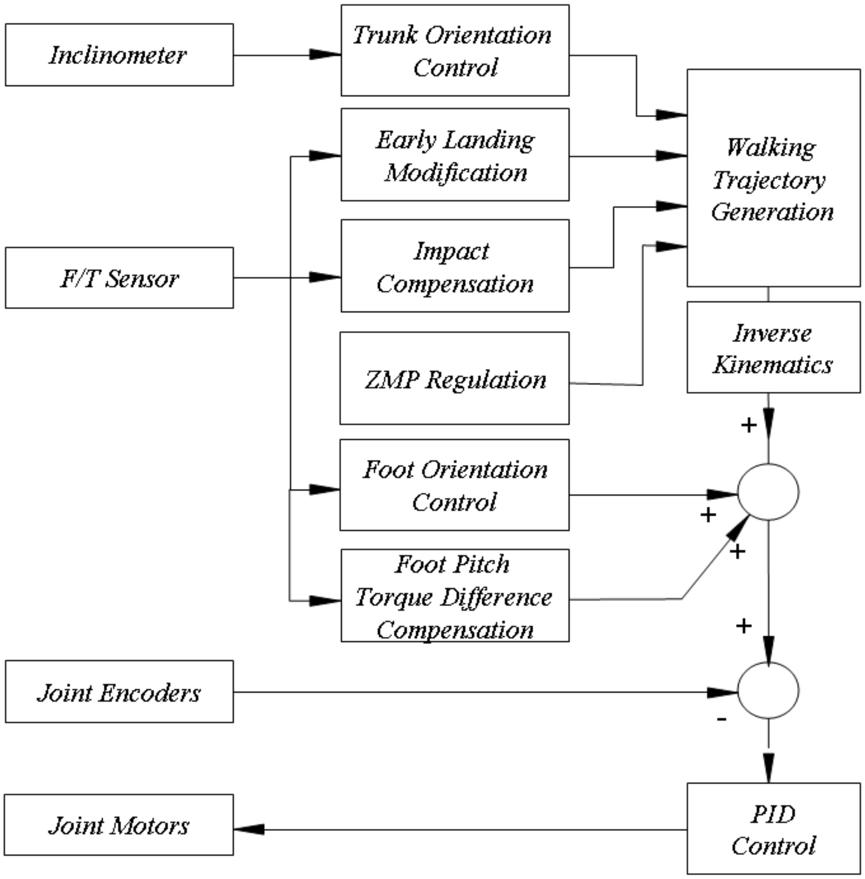


Figure 3.15: Walking control block diagram [49]

Chapter 4

4. COMPUTER VISION ALGORITHMS AND TERMINOLOGY

Achieving high level tasks on a humanoid robot requires sufficient information from the sensors. Images acquired by the vision systems (cameras) are providing high-level information about the environment. In this thesis, mainly Pose Estimation and Tracking problems are discussed. The robot must know the three dimensional location of the object to get it from the environment. Therefore 3D position and orientation of object is estimated with chosen pose estimation algorithm. After locating the object, robot must track it with its head, so that vision system can continue to estimate the position of object as robot walks.

4.1. Object Tracking Using HSV Color Space

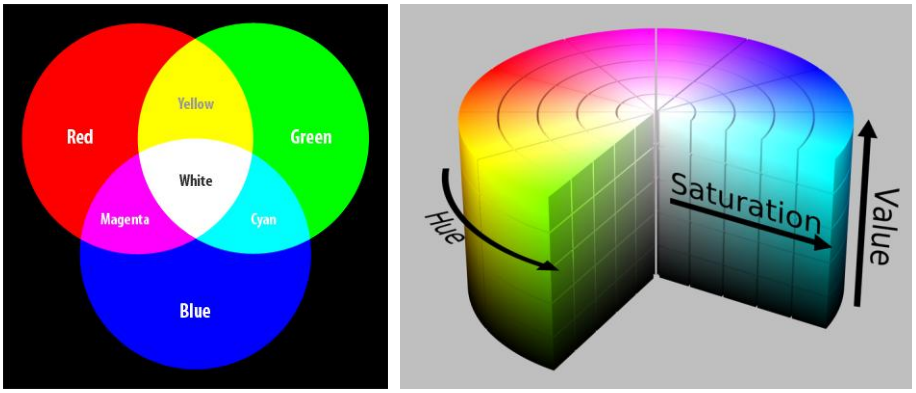
To be able to locate the object in 3D world, first of all object must be detected in 2D image. Using HSV color space and assuming the object has a square printed on it with specified color, object detection is achieved.

HSV has three channels: hue, saturation and value. Hue can be defined as a value which represents the similarity between an input color and main colors (red, green and blue). Saturation stands for the degree of difference between an input color and its brightness. Value is the grayscale value of a color (intensity). Grayscale images are formed by black and white colors (shades of gray).

HSV is a color model, which represents pixels in RGB color model in cylindrical coordinates. RGB stands for red channel, green channel and blue channel, where HSV stands for hue, saturation and value. The most important difference between these two color spaces can be described as HSV identifies the pixels by their hue (color), whereas RGB only gives information on separate channels of red green and blue(Figure 4.1(a)). That means color information must be calculated from these three channels. But in HSV space any color can be identified and marked on image simply by tresholding operation.

In HSV space colors are mapped to a cylinder (Figure 4.1 (b)). Hue channel can be expressed as an angle on circle side of the cylinder. It starts from red color (zero degree) and ends at red color again (360 degree). Saturation and value channels can be expressed as a line and corresponds to diameter and height of the cylinder respectively.

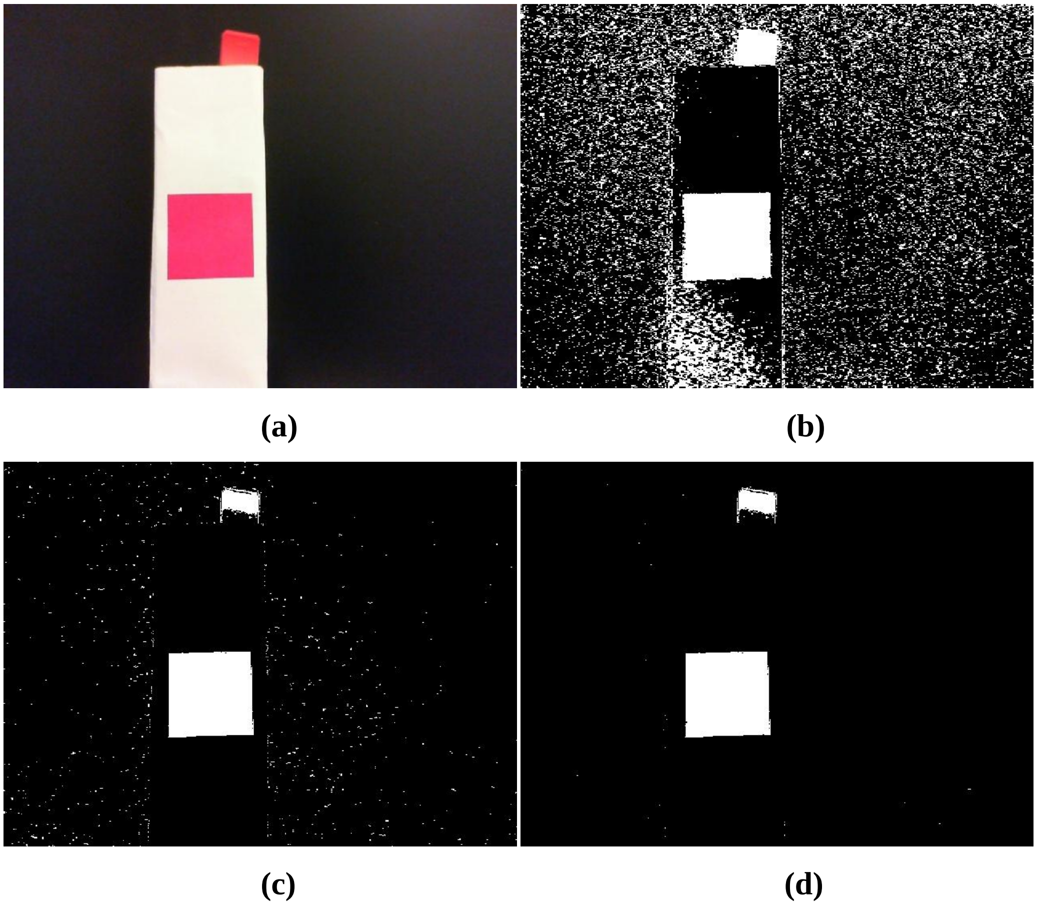
Vision systems use 8 bits to represent each separate channel. Therefore angles from hue channel (0^o-360^o) are mapped to 8 bits (0-255).



(a) **(b)**

Figure 4.1 (a) RGB color space representation, (b) RGB color space representation [80]

Any color can be separated from others in HSV color space by combination of hue and saturation channel. In the final system that is build for this thesis, user can click on a point in the acquired image to set the object color and saturation level within a range. Using these values, separating the colored region is simply by checking all pixels' hue and saturation values to see if they are in range. The pixels within the range will be labeled by their hue's and saturation's label in separate channels. After that two channels are reduced to one by bitwise or operation. The goal of these processes is to create a mask that will be used in corner detection. But these operations alone are not sufficient to create a mask. There will be some noise in the data therefore morphological operations are used to get a reliable mask. Figure 4.2 shows an example of the operations that is described so far.


(a) **(b)**

(c) **(d)**

Figure 4.2 Channels of image with red square on it. a) Original image b) Hue channel of original image in HSV representation c) Saturation channel of original image in HSV representation d) Binary image created with bitwise or operation of H and S channels.

4.2. Morphological Operations

Morphological operations are the most common binary image operations. They are filters which are convolved with binary image and give output depending on the operation [57]. They have a structuring element which determines the shape and size of the filter. The structuring element can be any shape and have any size. They can change the shape of the binary object or binary regions of the input binary image. Some of the most common operations are erosion, dilation opening and closing.

Erosion thins (shrinks) the binary object, on the other hand dilation thickens (grows) it. Opening and closing are combination of erosion and dilation and they are intended to be used as removing small parts of the binary image and smoothing the boundaries. For opening, input image is first eroded then dilated. Closing is the reverse of that operation; image is first dilated then eroded. Figure 4.3 shows the input and output images for these common morphological operations.

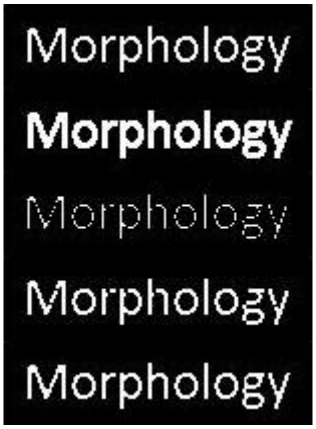
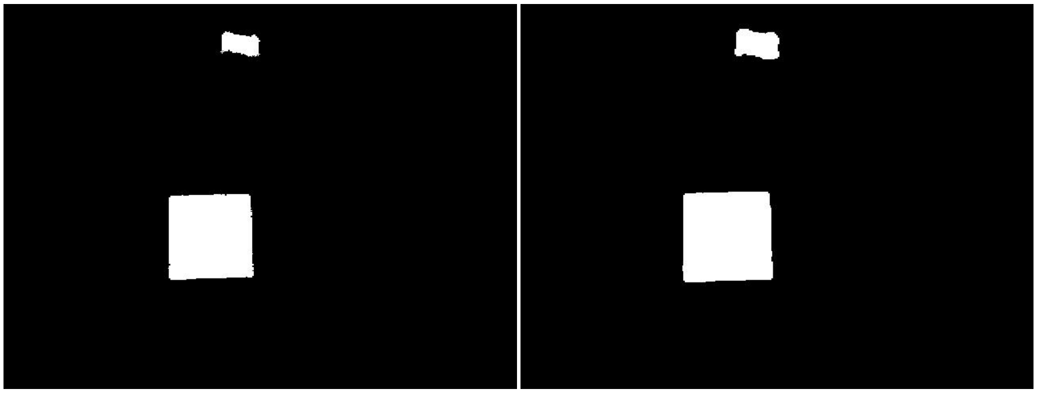


Figure 4.3 Common morphological operations. From top to bottom: original binary image, dilation, erosion, opening and closing operation.

To create a mask for corner detection closing and dilation operations are used after bitwise or operation of binary Hue and Saturation channels. The structuring elements for both operations are simple 3x3 box filter. Results are shown in Figure 4.4



(a) **(b)**

Figure 4.4 Opening and dilation morphological operations applied to binary image in Figure 4.2 (d).

Resulting image can be a mask for corner detection, but there is a catch. If there is more than one chosen colored object in the scene corner detector will give corners that are not in the interest region. To avoid this problem the biggest object should be detected and other objects should be removed from the mask. This solution also has its own issues, such as if there are two objects and the big one is not the selected object then results will be wrong. To overcome this it is assumed that there won't be any bigger object on the scene with the same color as chosen color.

Detecting the largest object on binary image is achieved by finding its contours and selecting the largest contour. This is done by using OpenCV version of [58].

In [58], finding contours of a binary image is achieved with border following algorithm. Border following is a technique which examines the connected components of ones and zeros to derive sequences of the image points from the borders of components. An example is shown in Figure 4.5.

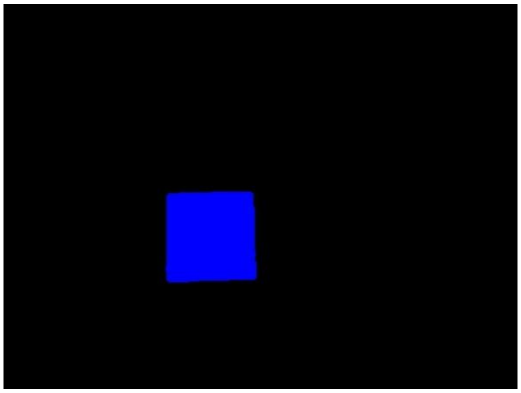


Figure 4.5 The largest contour is found and labeled using the image in Figure 4.4 (b).

4.3. Harris Corner Detection

Harris feature detection algorithm is a combined edge and corner detector. Algorithm uses intensity changes in image to detect if it is an edge or a corner. Edge can be defined as a contour in the image where the intensity values of points on the contour are same. Then corner can be defined as a point where there are two or more edges around its neighborhood with different directions.

[59] proposed a method based on [60], which is using local-correlation function and local maxima to achieve rotation invariant corner detector. [60] defined change of intensity for the shift [u, v] as,

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u,y+v) - I(x,y)]^2 \tag{4.1}$$

where $w(x,y)$ is the window function, $I(x+u,y+v)$ is shifted intensity and $I(x,y)$ is intensity. [60] used binary function as a window function where [59] used a Gaussian function (Figure 4.6),

$$w(x,y) = \exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right) \tag{4.2}$$

For small shifts E can be written as,

$$E(u,v) \cong [u,v]M \begin{bmatrix} u \\ v \end{bmatrix} \tag{4.3}$$

where M contains derivative of the image and 2x2 matrix,

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \tag{4.4}$$

Eigenvalues of M, namely λ_1 and λ_2 are proportional to the principal curvatures of the local auto-correlation function. Since they can be represented as an ellipse (Figure 4.7), and since an ellipse's eigenvalues does not change when ellipse is rotated; the eigenvalues form a rotationally invariant descriptor of M.

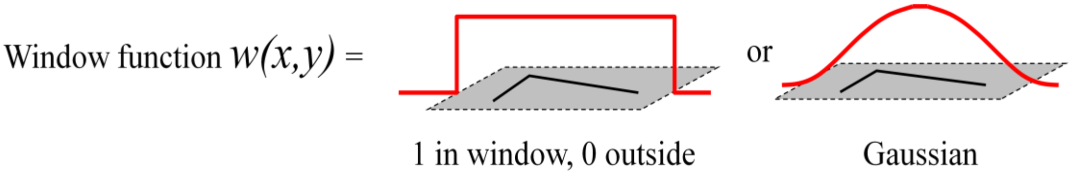


Figure 4.6 Binary and Gaussian window functions [59]

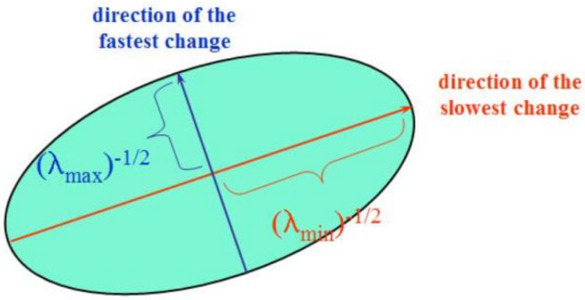

direction of the fastest change
direction of the slowest change
$(\lambda_{max})^{-1/2}$
$(\lambda_{min})^{-1/2}$

Figure 4.7 Uncertainty ellipse corresponding to an eigenvalue analysis of auto-correlation matrix [59]

Classification of image points can be done once the eigenvalues of M is found. If both eigenvalues are large and their values are similar then it can be said that the image point belongs to corner region. If one of them has a significantly higher value than the other, then the image point is belongs to edge region. If the image point does not belong to either region then it belongs to flat region (Figure 4.8). To measure quality of the corners [59] used a corner response function,

$$R = \det(\mathrm{M}) - \mathrm{k}\,\mathrm{trace}(\mathrm{M})^2 \tag{4.5}$$

where $\det(\mathrm{M}) = \lambda_1\,\lambda_2$, $\mathrm{trace}(\mathrm{M}) = \lambda_1 + \lambda_2$ and k is empirical constant and equals to 0.06. Algorithm calculates the R values from the image and takes the points of local maxima of R as a corner point. An example image is shown in Figure 4.9.

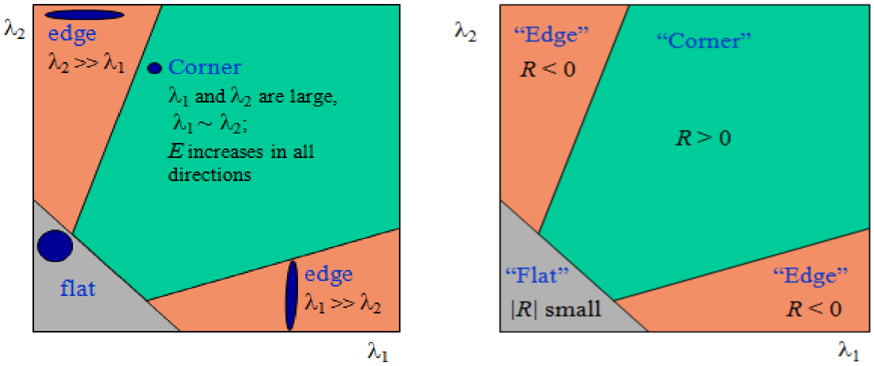


Figure 4.8 On the left side image point classification based on the eigenvalues of M. On the right side image point classification based on corner response [59]

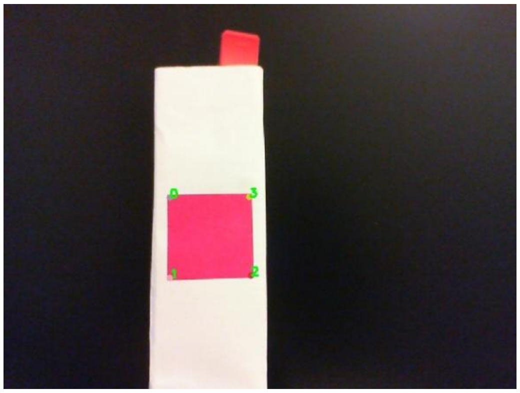


Figure 4.9 Corners of a red square labeled from zero to four.

4.4. Pinhole Camera Model

Pinhole Camera Model is a mapping of the points in 3D to 2D [61]. The model assumes an ideal scenario where ideal pinhole camera is used. Ideal pinhole camera does not contain any lenses and its aperture is described as a point. Scene points are projected to image plane by passing through center of focus of the camera. This model simplifies the mathematical relationship of the scene points and image points. Image obtained from pinhole camera model is reversed version of original points. Forward pinhole camera model is another variation of pinhole camera model where the image plane lies in front of center of focus. In forward pinhole camera model, images are not obtained reversely.

Figure 4.10 and Figure 4.11 shows pinhole camera model and forward pinhole camera model respectively. f corresponds to focal length. Focal length is the distance between center of focus and image plane. p is scene points and can be extended to $p = (X, Y, Z)$. $\boldsymbol{x}$ is image point and it has two components, $\boldsymbol{x} = (x, y)$. O is the center of focus.

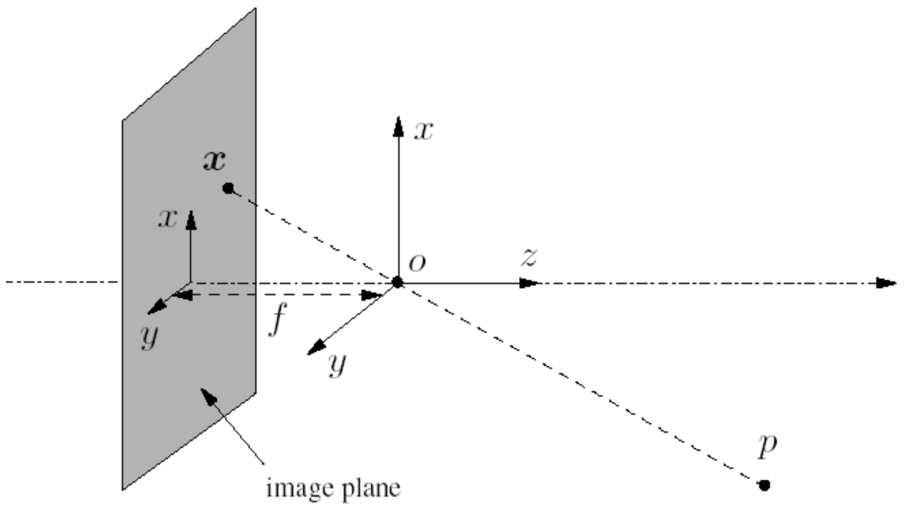


Figure 4.10 Pinhole Camera Model

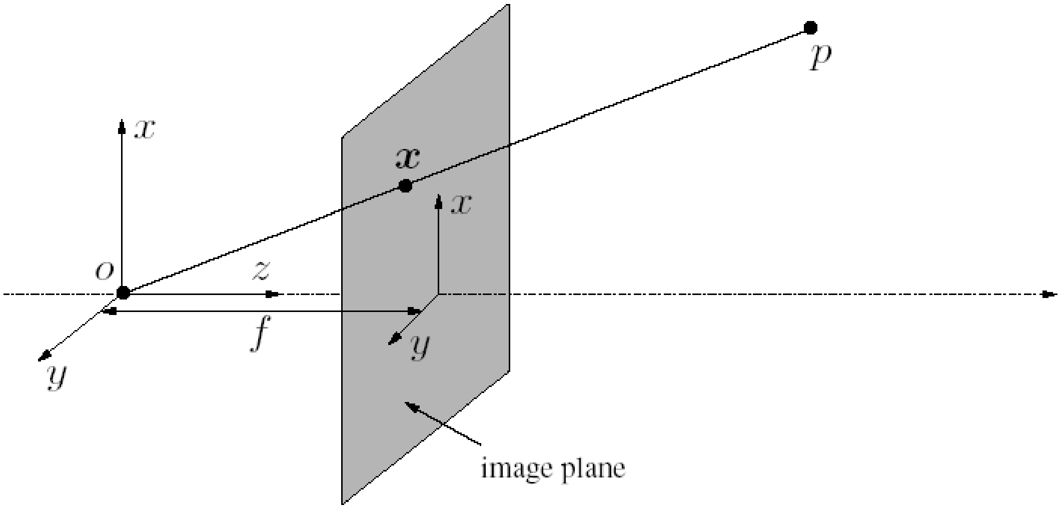


Figure 4.11 Forward Pinhole Model

$\boldsymbol{x}$ can be calculated by geometric approach,

$$x = -f\frac{X}{Z} \qquad (4.6)$$
$$y = -f\frac{Y}{Z}$$

For forward pinhole camera model,

$$x = f\frac{X}{Z} \qquad (4.7)$$
$$y = f\frac{Y}{Z}$$

If focal length is not used in this calculation, resulting coordinates are called normalized coordinates and equal to:

$$x_n = \frac{X}{Z} \qquad (4.8)$$
$$y_n = \frac{Y}{Z}$$

Equation 4.7 can also be expressed as,

$$\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix} = \frac{f}{Z}\begin{bmatrix} X \\ Y \end{bmatrix} \qquad (4.9)$$

Using homogenous coordinates, Equation 4.9 can be written as,

$$Z\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad (4.10)$$

3x4 matrix from Equation 4.10 can be separated to two terms,

$$\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.11}$$

Where first 3x3 matrix on Equation 4.11 corresponds to camera matrix,

$$K_f = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.12}$$

And 3x4 matrix on Equation 4.11 corresponds to projection matrix,

$$\Pi_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.13}$$

Rigid body transformation of the points in 3D can be shown as,

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix}, \\ g = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \tag{4.14}$$

Then by combining Equations 4.10, 4.12, 4.13 and 4.14; image coordinates in metrics takes the form,

$$\lambda \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \\ = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix} \tag{4.15}$$

After finding image coordinates in metrics, the points should be mapped to pixels. To be able to map the points there should be parameters that scales the x and y axis distances on image coordinates in metrics to pixel coordinates. Those parameters are s_x and s_y and called skew parameters. Commercial cameras are not perfect and that can cause the angle between x and y axes of the photoreceptors of the camera to be distorted, and not become perfect 90°. s_θ is representing this imperfection. But s_θ is assumed to be zero in most computer vision application and also in this thesis its value is taken as zero. c_x and c_y are the distance in 2D to optical center (center of image). Mapping image coordinates in metrics to pixel coordinates can be expressed as,

$$\boldsymbol{x'} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & s_\theta & c_x \\ 0 & s_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{4.16}$$

When the Equations 4.15 and 4.16 are merged, final pinhole camera model becomes,

$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & s_\theta & c_x \\ 0 & s_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \\ 1 \end{bmatrix} \tag{4.17}$$

4.5. Radial Distortion

The optic lenses that are used in cameras are not manufactured perfectly and they cause the distortion effect on the captured images. The straight lines in 3D scene are not straight on the captured image due to radial distortion. Distortion effect can be observed the most for the rays which are passing through the edge of the lenses.

Radial distortion can be modeled and the image points can be corrected. Common radial distortion models are shown in Figure 4.12. Brown’s distortion model is the most common mathematical distortion model [62]. In this model, a polynomial function is fit to pixels depending on their distance to optical center. With this function distorted coordinates (x_d, y_d) are calculated as follows,

$$x_d = x' + (x' - c_x)(k_1 r^2 + k_2 r^4 + \cdots) + \left[p_1(r^2 + 2(x' - c_x)^2) + 2p_2(x' - c_x)(y' - c_y)\right](1 + p_3 r^2 + \cdots) \tag{4.18}$$

$$y_d = y' + (y' - c_x)(k_1 r^2 + k_2 r^4 + \cdots) + \left[p_1\left(r^2 + 2(y' - c_y)^2\right) + 2p_2(x' - c_x)(y' - c_y)\right](1 + p_3 r^2 + \cdots). \tag{4.19}$$

And r is defined as,

$$r = \sqrt{(x' - c_x)^2 + (y' - c_y)^2} \tag{4.20}$$

Radial distortion parameters are $k_1 \dots k_n$ and tangential distortion parameters are $p_1 \dots p_n$. For practical applications in computer vision, usually first two terms of these parameters are enough to clean the distortion for most commercial optic lenses.

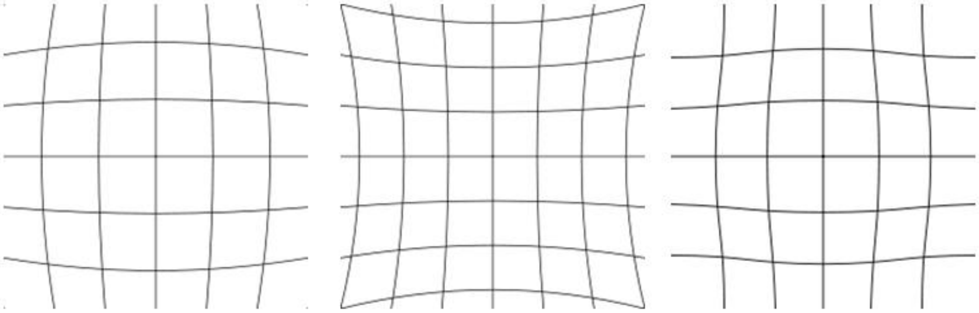


Figure 4.12 Types of radial distortions. Barrel, pincushion and mustache distortion are shown respectively (from left to right) [81]

4.6. Camera Calibration

Camera calibration is the process to find intrinsic and extrinsic parameters of camera. Intrinsic parameters of camera are focal length, skew parameters ($s_x,\ s_y$ and s_θ), center of image (c_x and c_y) and radial distortion coefficients. Extrinsic parameters of camera are rotation and translation of the scene points relative to the camera frame. These parameters can be estimated by series of images (or just one image) of an object with known geometry.

The most common camera calibration method is Zhang's camera calibration algorithm [63]. Zhang's method is robust and flexible, which means it can adapt to any commercial and industrial camera. Zhang used only one plane for the calibration. Although two planes would be more precise, one plane reduces the cost and it is easier to build.

Calibration requires an object with known geometry and it is usually chosen as chess pattern. The fixed points on the chest patterns can be found by any corner detection algorithm.

Images taken with random orientation of calibration pattern are used to compute homography between the images. Homography is a linear map in projective space. Homography matrix can transform points from first image to second image. Because the transform is linear, map is invertible (onto and one-to-one). The transform is generally 2D to 2D and it preserves the lines. But 3D to 2D transforms which preserves the lines are also accepted as homography. An example to that is perspective projection (which preserves lines). Homography matrices are used in panoramic images, navigation applications and augmented reality applications.

For n different oriented images, n different homography matrices are estimated. After getting homography matrices intrinsic parameters can be computed analytically. From intrinsic parameters, both extrinsic and radial distortion parameters can be computed. To improve the results, Zhang used maximum likelihood estimation (MLE). MLE estimates the parameters of a given statistical model.

Mathematically,

$$\lambda \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{4.21}$$

H is the 3×3 homography matrix which relates 3-D scene points to 2-D image points up to a scale. H can be expressed as,

$$H = [h_1 \quad h_2 \quad h_3] = \lambda \begin{bmatrix} s_x & s_\theta & c_x \\ 0 & s_y & c_y \\ 0 & 0 & 1 \end{bmatrix} [r_1 \quad r_2 \quad T] \tag{4.22}$$

$$A = \begin{bmatrix} s_x & s_\theta & c_x \\ 0 & s_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4.23}$$

where r_1 and r_2 are the first and second column of rotation matrix R. Using the knowledge that two columns of a rotation matrix must be orthonormal to each other, constraints on the intrinsic parameters are defined,

$$\begin{aligned} h_1^T A^{-T} A^{-1} h_2 &= 0 \\ h_1^T A^{-T} A^{-1} h_1 &= h_2^T A^{-T} A^{-1} h_2 \end{aligned} \tag{4.24}$$

A homography matrix has 8 DoF and there are 6 extrinsic parameters (3 for rotation and 3 for translation), therefore 2 constraints on the intrinsic parameters are obtained from one homography matrix. Then analytic solution can be written as,

$$B = A^{-T} A^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix} \tag{4.25}$$

B is symmetric matrix and can be defined by 6D vector,

$$b = [B_{11} \quad B_{12} \quad B_{22} \quad B_{13} \quad B_{23} \quad B_{33}] \tag{4.26}$$

$h_i = [h_{i1} \quad h_{i2} \quad h_{i3}]^T$ is defined as the *i*^th^ column vector of H,

$$h_i^T B h_j = v_{ij}^T b$$
$$v_{ij} = \begin{bmatrix} h_{i1}h_{j1} & h_{i1}h_{j2} + h_{i2}h_{j1} & h_{i2}h_{j2} & h_{i3}h_{j1} + h_{i1}h_{j3} & h_{i3}h_{j2} + h_{i2}h_{j3} & h_{i3}h_{j3} \end{bmatrix}^T \tag{4.27}$$

For a homography matrix, constraint Equation 4.24 can be rewritten as,

$$\begin{bmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{bmatrix} b = 0 \tag{4.28}$$

If there are *n* images of the model plane, by stacking Equation 4.28,

$$Vb = 0 \tag{4.29}$$

where *V* is a $2n \times 6$ matrix. Unique solution can be obtained if $n \geq 3$. *b* can be estimated using eigenvector of $V^T V$ associated with the smallest eigenvalue. With estimated *b*, camera intrinsic matrix A can be estimated. From 4.22,

$$\begin{gathered} r_1 = \lambda A^{-1} h_1 \\ r_2 = \lambda A^{-1} h_2 \\ r_3 = r_1 \times r_2 \\ T = \lambda A^{-1} h_3 \\ \lambda = 1/\| \, A^{-1} h_1 \| = 1/\| \, A^{-1} h_2 \| \end{gathered} \tag{4.30}$$

Obtained solution is refined through (MLE),

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \left\| x'_{ij} - \widehat{m}(A, R_i, T_i, X_j) \right\| = 0 \tag{4.31}$$

$\hat{m}(A, R_i, T_i, X_j)$ is the projection of point X_j in image i. This nonlinear minimization problem is solved by *Levenberg-Marquardt* (LM) algorithm [67].

In the implementation of this thesis, OpenCV version of [64] is used for camera calibration (Zhang's method).

4.7. Pose Estimation

Pose estimation is the problem of finding the transformation between camera frame and the object frame. Transformation consists of rotation and translation, which states that pose estimation can be defined as the problem of finding camera extrinsic parameters relative to target object or scene. Generally to solve this problem, correspondence between 3D scene and 2D image must be arranged.

In this thesis's implementation, OpenCV implementation of CamPoseCalib (CPC) is used for solving pose estimation problem [65]. CPC approaches the problem as a non-linear least squares problem. It uses correspondence between 3D and 2D points to calculate reprojection error and make iteration until the error converges to zero or stays lower than a threshold (generally very low value).

Algorithm starts with an initial estimate of pose and it reprojects the 3D points using these pose and calculates the reprojection error. If error is not under the predetermined threshold then it continues to estimate new parameters for pose and first cycle is repeated with these new pose. It searches until a suitable pose is found.

In mathematically, algorithm can be written as,

$$\hat{\theta} = argmin_{\theta} \sum_{i=1}^{m} \left(r_i(\theta)\right)^2 \tag{4.32}$$

m is defined as the number of correspondence between 3D scene and 2D image. $r_i(\theta)$ are the residue functions which represent the reprojection error $d = r_i(\theta)^2 = r_x^2 + r_y^2$. In [65], $\theta = \left(\theta_x, \theta_y, \theta_z, \theta_\alpha, \theta_\beta, \theta_\gamma\right)^T$ is defined as pose vector containing three translation and three rotation parameters. Projected point from the obtained pose is defined as $m'(X_i, \theta)$ where X_i is the 3D scene point. Then residue functions can be defined as,

$$r_i(\theta) = m'(X_i, \theta) - \boldsymbol{x}' \tag{4.33}$$

$\boldsymbol{x}'$ is the image coordinates in pixels which is described in section 4.1.6. The projection can be rearranged using camera intrinsic parameters as follows,

$$m'(X_i, \theta) = \begin{pmatrix} s_x \dfrac{m_x(X, \theta)}{m_z(X, \theta)} + c_x \\ s_y \dfrac{m_y(X, \theta)}{m_z(X, \theta)} + c_y \end{pmatrix} \tag{4.34}$$

In Equation 1.23, $m(X, \theta) = \left(m_x, m_y, m_z\right)^T$ is the rigid motion in 3D and equals to,

$$m(X, \theta) = \left(\theta_x, \theta_y, \theta_z\right)^T + R_x(\theta_\alpha) R_y\left(\theta_\beta\right) R_z\left(\theta_\gamma\right) X \tag{4.35}$$

The solution of the optimization problem that is found by *(LM)* algorithm is shown in Equation 4.36. *I* corresponds to identity matrix and *J* is the Jacobian with the partial derivatives of the residue functions. More detailed derivation can be found in [66]. The Levenberg-Marquardt algorithm is explained in detail on [67].

$$\Delta\theta = -(J^T J + \lambda I)^{-1} J^T r(\theta_t) \tag{4.36}$$

For the matrix $J^T J$ to be invertible, det $(J^T J)$ must be > 0. To achieve this property, at least three correspondences are needed. Each point carries two equations and there are six unknowns in the system. In OpenCV implementation, the radial distortion coefficients are used to find undistorted image so that estimation can be done more precisely.

Chapter 5

5. GAZE CONTROL

In the first section of this chapter visual servo control (also called visual servoing) is described in detail. In the second chapter, a gaze control law is derived for SURALP.

Gaze control can be described as to track an object with humanoid robot's *gaze*. Gaze can be described as the direction of image plane normal in positive direction. Usually Humanoid robots should achieve this task by rotating their head as the capability to rotate cameras individually is rare. Successful implementation of gaze control algorithms on humanoid robots are reported in [68-71]. In these works, visual servoing was used with adjustments to relative task. To be able to describe gaze control, firstly visual servoing is described.

5.1. Visual Servo Control

Visual servo control or visual servoing refers to controlling a robot (or a motor) with computer vision data. There are two main configurations for visual servoing, camera can be mounted on end manipulator of the robot (eye-in-hand) or camera can be mounted to a position where it can observe the end manipulator of the robot (eye-to-hand).

The origin of visual servoing was [72]. Remarkable works and tutorials on visual servoing are presented [73-78]. In this section terminology in [77] is used.

The goal of visual servoing is the same with any other servoing problem, minimizing the error function. The most common error function is defined as follows,

$$e(t) = s(m(t), a) - s^* \tag{5.1}$$

$m(t)$ is a vector that contains image measurements (interest points of the image like corners, edges, pose of the object, …etc.). From this image measurements visual feature vector $s(m(t), a)$ can be computed, where a is a set of parameters that has additional information about the system. This additional information can be camera intrinsic parameters, 3D model of objects… etc. s^* is the desired features vector. Camera is considered to be 6 DoF in 3D Euclidian space. The spatial velocity of the camera can be written as $\mathrm{v}_c = (v_c, w_c)$, where v_c is the instantaneous linear velocity part and w_c is the instantaneous angular velocity of the camera frame. Derivative of the visual feature vector can be linked to camera spatial velocity by

$$\dot{s} = L_s \mathrm{v}_c, \tag{5.2}$$

where $L_s \in \mathbb{R}^{kx6}$ (k is the dimension of s). L_s is the *interaction matrix*, it can be called *feature Jacobian* too. Merging Equations 5.1 and 5.2 derivative of the error is obtained,

$$\dot{e} = L_e \mathrm{v}_c. \tag{5.3}$$

In Equation 5.3 $L_e = L_s$. Generally obtained camera velocity, v_c, is given to the robot as a controller input. To obtain v_c, L_e must be inverted, but not all interaction matrices are invertible. Therefore Moore-Penrose pseudo-inverse of L_e is written as follows,

$$L_e^+ = ({L_e}^T L_e)^{-1} {L_e}^T \tag{5.4}$$

Note that when L_e is full rank (6) and k is equal to 6, then L_e becomes invertible and ${L_e}^{-1}$ can be directly used. $\dot{e} = -\lambda e$ is used to reduce the error exponentially. This choice also ensures the system is globally asymptotically stable when the following condition is met

$$L_e L_e^+ > 0 \tag{5.5}$$

Using. $\dot{e} = -\lambda e$, Equation 5.3 becomes

$$\mathrm{v}_c = -\lambda \widehat{L_e^+} e \tag{5.6}$$

The symbol $\widehat{L_e^+}$ shows that it is an approximation, as in real application L_e^+ cannot be computed precisely. After finding camera velocity, it can be fed to robot velocity controller.

There are three main kind of visual servo control, namely Image Based Visual Servoing (IBVS), Position Based Visual Servoing (PBVS) and Hybrid Visual Servoing.

IBVS uses an interaction matrix which contains the parameters retrieved from a feature extractor. In other words, IBVS drives the motors by using the information on image plane. It does not estimate any 3D information (pose estimation and 3D reconstruction are not used). PBVS's interaction matrix includes 3D pose information of selected feature and its control scheme is usually based on pose estimation algorithm. Hybrid Visual Servoing is the mixture of IBVS and PBVS. The common approach is to add both interaction matrices from IBVS and PBVS to get hybrid interaction matrix.

In this thesis's implementation IBVS control is used. In next section, IBVS control scheme is explained.

5.1.1. Image Based Visual Servoing

The projection of 3D points ($\mathbf{X} = (X, Y, Z)$) in to 2D image plane is, [77],

$$\begin{aligned} x_n &= \frac{X}{Z} = (x' - c_x)/fs_x \\ y_n &= \frac{Y}{Z} = (y' - c_y)/fs_y \end{aligned} \tag{5.7}$$

Parameters are the same with pinhole camera model section; $\boldsymbol{x_n} = (x_n, y_n)$ is normalized image coordinates, $\boldsymbol{x'} = (x', y')$ are pixel coordinates, s_x, s_y scale factors of x axis and y axis respectively, c_x and c_y are distance to center of image and f is focal length. [77] considered $s = \boldsymbol{x_n} = (x_n, y_n)$ as visual feature vector. Therefore only image plane is used. Time derivative of $\boldsymbol{x_n} = (x_n, y_n)$ equals to,

$$\begin{aligned} \dot{x}_n &= \frac{\dot{X}}{Z} - \frac{X\dot{Z}}{Z^2} = (\dot{X} - x_n\dot{Z})/Z \\ \dot{y}_n &= \frac{\dot{Y}}{Z} - \frac{Y\dot{Z}}{Z^2} = (\dot{Y} - y_n\dot{Z})/Z \end{aligned} \tag{5.8}$$

Velocity of the 3D point can be expressed in terms of camera spatial velocity as follows,

$$\dot{\mathrm{X}} = -v_c - w_c \times \mathrm{X} \Leftrightarrow \begin{cases} \dot{x} = -v_x - w_y Z + w_z Y \\ \dot{Y} = -v_y - w_z X + w_x Z \\ \dot{Z} = -v_z - w_x Y + w_y X. \end{cases} \tag{5.9}$$

Merging Equation 5.8 and 5.9, obtained equation is,

$$\begin{cases} \dot{x}_n = -v_x/Z + x_n v_z/Z + x_n y_n w_x - (1 + {x_n}^2) w_y + y_n w_z \\ \dot{y}_n = -v_y/Z + y_n v_z/Z + (1 + {y_n}^2) w_x - x_n y_n w_y - x_n w_z \end{cases} \tag{5.10}$$

Equation 5.10 can be written as

$$\dot{\boldsymbol{x}}_{\boldsymbol{n}} = L_x \mathrm{v}_c \tag{5.11}$$

Where L_x is the interaction matrix that related to x and it is equal to

$$L_x = \begin{bmatrix} -1/Z & 0 & x_n/Z & x_n y_n & -(1 + {x_n}^2) & y_n \\ 0 & -1/Z & y/Z & 1 + {y_n}^2 & -x_n y_n & -x_n \end{bmatrix} \tag{5.12}$$

To be able to compute the interaction matrix L_x, Z must be known. Every point in image plane contains two information parameters. 6 DoF camera velocity vectors contains 6 unknown, therefore three points are sufficient to estimate these 6 unknowns. For every point L_{x_i} can be computed and by stacking these interaction matrices to form final interaction matrix. For three points,

$$L_x = \begin{bmatrix} L_{x_1} \\ L_{x_2} \\ L_{x_3} \end{bmatrix} \tag{5.13}$$

is obtained. With the equations derived in this section IBVS control can be achieved. In the next section, IBVS for gaze estimation will be discussed.

5.2. Derivation of IBVS for Gaze Control

Gaze control's goal is to control the neck of humanoid robot according to visual data. Humanoid robot SURALP has a neck with 2 DoF. The goal of gaze control in this thesis context is to make sure that the object is always in the middle of the image plane. Therefore when SURALP is walking, the object will stay in line of sight of the camera.

The neck can rotate among x and y axis and it cannot move on the any other direction without whole robot moves. When Z is held constant the Equation 5.8 becomes as follows,

$$\begin{cases} \dot{x}_n = \dot{X}/Z \\ \dot{y}_n = \dot{Y}/Z. \end{cases} \tag{5.14}$$

When Equation 5.9 is injected into 5.13, it becomes,

$$\begin{cases} \dot{x}_n = -v_x/Z - w_y + y_n w_z \\ y_n = -v_y/Z - w_z x_n + w_x. \end{cases} \tag{5.15}$$

Only w_x and w_y can be controlled therefore the other parameters are assumed to be zero. Then Equation 5.14 becomes,

$$\begin{cases} \dot{x}_n = -w_y \\ \dot{y}_n = w_x. \end{cases} \tag{5.16}$$

It can be re-written as

$$\begin{gathered} \begin{bmatrix} \dot{x}_n \\ \dot{y}_n \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} w_x \\ w_y \end{bmatrix} \\ L_x = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}. \end{gathered} \tag{5.17}$$

The interaction matrix is constant and invertible. For gaze control only one image point is enough as it gives two equations. Simulation results are shown in Figure 5.1 and Figure 5.2. The implementation on humanoid robot is successful and results are shown in Figure 5.3. Robot can track the object with its head successfully in real-time.

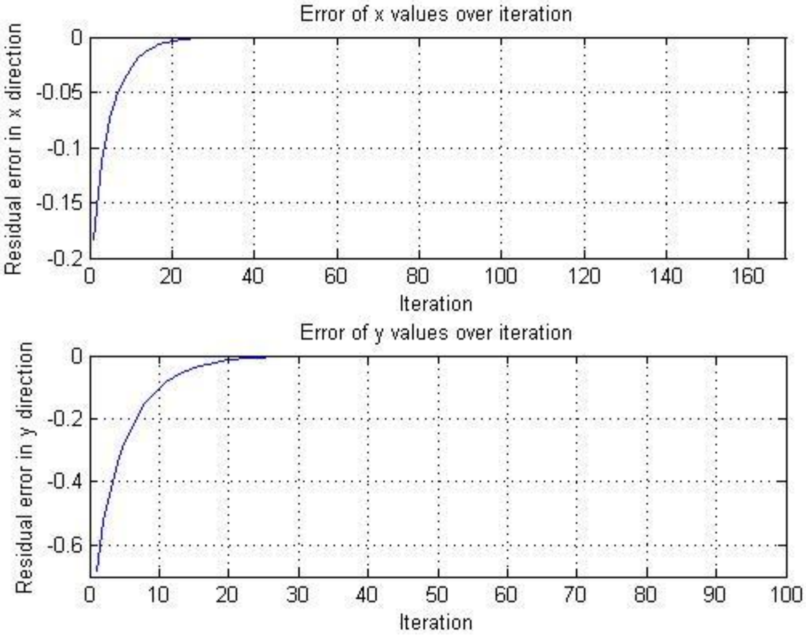


Figure 5.1 Residual errors from 2 DoF gaze control simulation

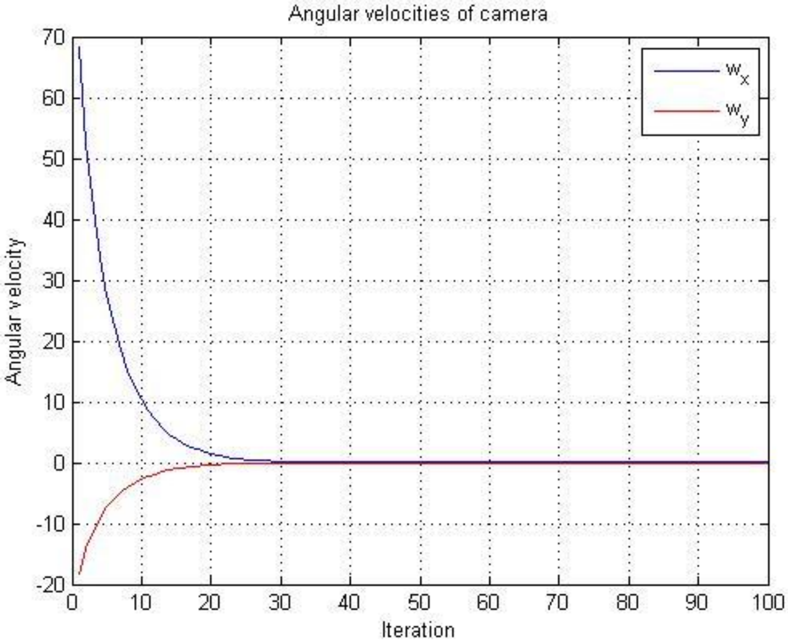


Figure 5.2 Angular velocities calculated in gaze control simulation

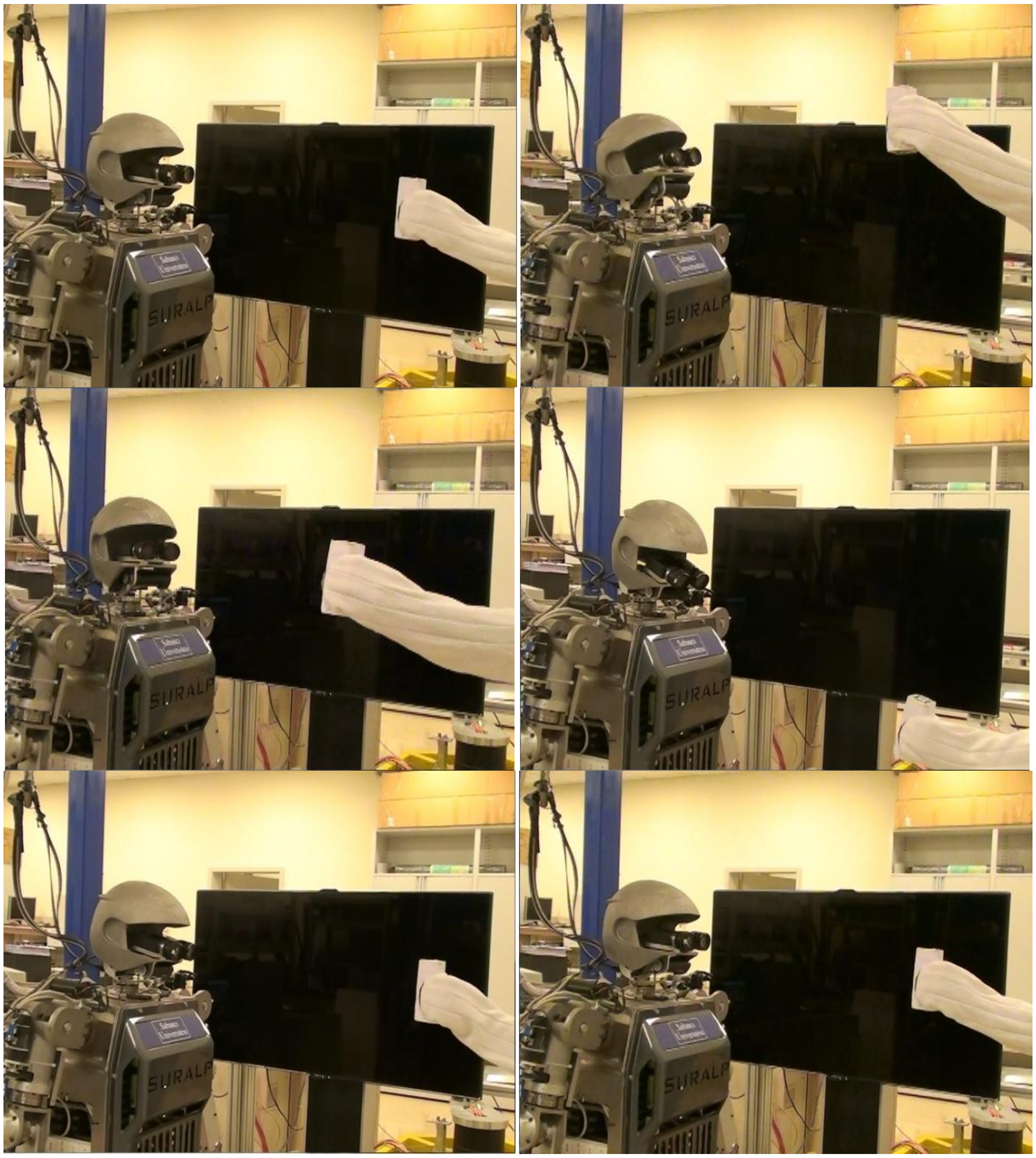


Figure 5.3 Gaze control implementation on SURALP. Robot follows an object with red square marker fixed on it.

Chapter 6

6. FINAL SYSTEM STRUCTURE

In this thesis, SURALP's hardware hierarchy is altered. A new computer is added to the system for computer vision calculations. Main computer is control computer (Laptop PC on Figure 3.5) of SURALP, which runs dSpace Control Desk to establish communication with servo drivers and give commands to SURALP. The secondary computer is used for image acquisition, image processing and computer vision algorithms (this computer will be called Vision Computer). The connection between two computers is established by simple socket programming application. In this chapter, overall system workflow is described in detail.

6.1. Computer Vision Algorithms

Camera calibration is done in offline stage and parameters obtained are saved for later use. First, computers should be connected with each other.

After establishing the connection between two computers, system asks user to click on the object from Vision Computer. After this manual operation, system will work automatically. Actually this step can be automatic but it requires the color information of the object beforehand and must be set offline. This option makes the system more flexible and different colored objects can be used in runtime.

The object that is chosen to be target object is a white rectangular prism with a red square on one side. Normally, object recognition and object detection without markers should be used for commercial uses as the assumptions made in indoor artificial experiment site will not hold on outdoor environment. But it is not in scope of this thesis.

Image acquired by the cameras are converted to HSV space. When user clicks on the object, hue and saturation values of the object is stored. This stored values are used to set thresholds for the segmenting the object from image. Upper and lower thresholds are created by threshold offsets in both hue and saturation channels of image. The lightning of the environment are assumed to be constant, as experiments are done in indoor settings with constant lamination on the scene. Normally with lamination change, saturation level of the object should change and detection of object cannot be done.

Binary images obtained from threshold operation are merged with bitwise multiplication operation. This yields a binary image with object on it (object pixels are ones and others are zeros). But due to noise this image is not yet usable. To clean the small parts (mainly noise) morphological operator “opening” is used with 3x3 structural elements. Opening operation cleans the noise but also thins the object and can create little holes as the object’s region is not fully filled with ones. Therefore dilation morphological operator is used to fill those holes and to ensure the red square’s corners are included in this created mask.

These operations are not sufficient to separate the object from other similar colored background objects. In the background of the scene it is assumed that there won’t be any bigger objects with similar color to target object. With this assumption, biggest object can be found and identified as the target in the image. Object contours are found by the OpenCV function “findContours()”. Contours retrieved from the algorithm can be sorted by their size and biggest one can be chosen. When the biggest contour is filled with ones, final form of the mask matrix is obtained.

The mask matrix is used in corner detection algorithm, which results increased performance due to less pixels to check and preventing the false corners that can be part of the background. After this step OpenCV implementation of Harris Corner Detection is used to find objects corner. Only four corners with highest response are taken and labeled as corners.

Determining pose between camera and object requires solving the correspondence problem as stated in Chapter 4 Section 4.7. The object is assumed to be placed on a table or any other flat surface which is parallel to the ground. This assumption enables solving correspondence problem automatically. Corners are labeled from one to four with respect to their position to the center point of the object and they are matched with object model points. Object model is defined as a square with the same metric values as original object.

Pose estimation method that is discussed in chapter 4.7 is used for estimating position and orientation of the object. Four corner points of object is used with corresponding object model points. After obtaining pose information, corners are projected to image using camera parameters that is obtained from camera calibration matrix. Results are shown in Figure 6.1.

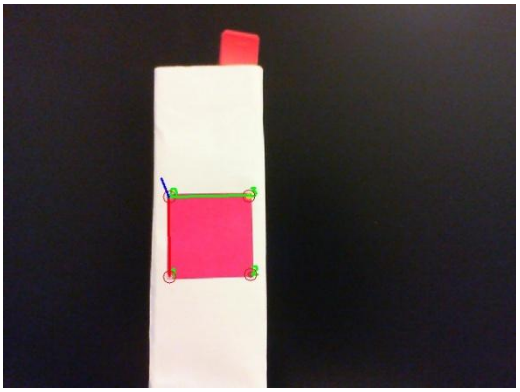


Figure 6.1 Object frame is drawn using estimated pose information

Vision computer sends the estimated pose information and object center point to control computer of SURALP.

6.2. Object Manipulation Algorithm

When control computer receives data from vision computer, it starts gaze control with arrived data. Then it calculates the translation vector from SURALP'S pelvis frame to object. The raw position data that comes from pose estimation algorithm cannot be used for object manipulation. It must be calculated using the neck angles and transformation matrices between camera frame and pelvis frame. To obtain this transformation matrix, Denavit-Hartenberg table is formed. Figure 6.2 shows the frame assignments and Table 6.1 is the corresponding Denavit-Hartenberg table.

Table 6.1 Denavit-Hartenberg parameters for neck to pelvis frame of SURALP

	a	α	d	θ
Link 1	0	90^o	0	θ_1^o
Link 2	0	0^o	0	θ_2^o

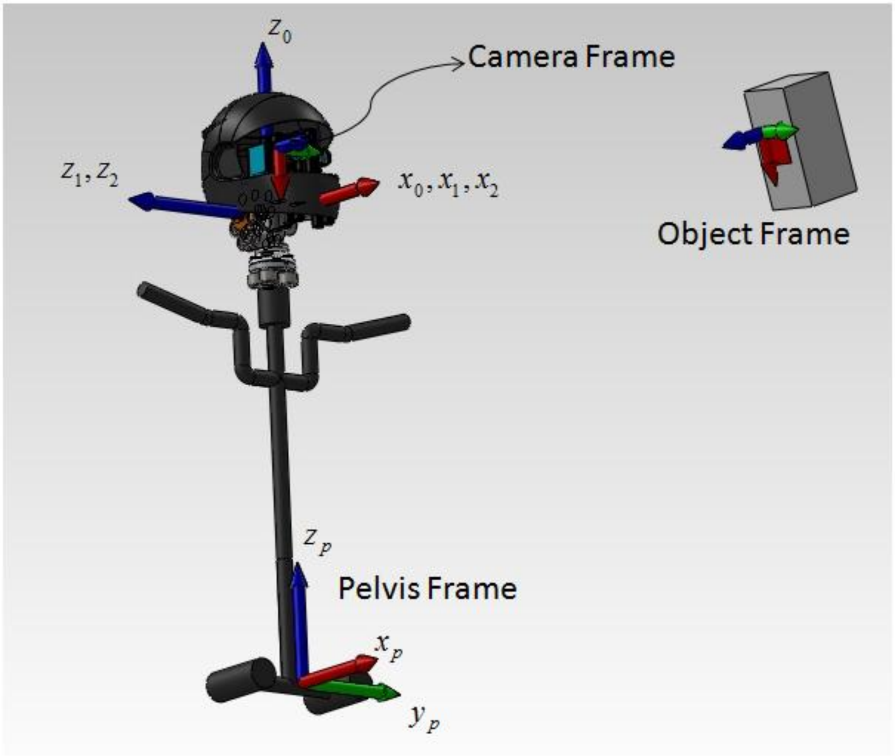


Figure 6.2 Denavit-Hartenberg frame assignments for neck to pelvis frame of SURALP

Object coordinates that are described in the pelvis frame can be used for navigation and object manipulation. Figure 6.2 shows SURALP's pelvis frame and object frame. Calculated object coordinates can be used to calculate r curvature value, which allows SURALP to turn left or right. Then SURALP can maintain itself to such a position, where it can get the object. Object should be located in the workspace of SURALP's right arm. In this thesis, object manipulation is done by right hand of SURALP.

Image quality restricts the corner detection results therefore pose estimation can only be done within 1.5 meters radius from the SURALP's head. Because of this restriction, radius of circular arc path r_c is fixed to minimum value. With sharper turns, SURALP can reach the desired point in shorter paths. But experiments show that with such a short distance, SURALPs workspace is highly limited.

When the object enters the workspace of SURALPs right arm, navigation is stopped. Robot decides if it can grab the object by a function which solves the inverse kinematics of robots 6 DoF arm. If the results are real numbers, then object can be graspable. Object's 3D position information is used to compute inverse kinematics for the right arm. Then SURALP reaches for the object and grabs it.

Figure 6.3shows the frames from the experiment video. SURALP successfully navigates through object and grabs it. Some experiments failed because the object is located outside of the SURALP. On those experiments, SURALP navigates through the object, tries to correct its body orientation and position to take object but the path ends before it can achieve the correct pose.

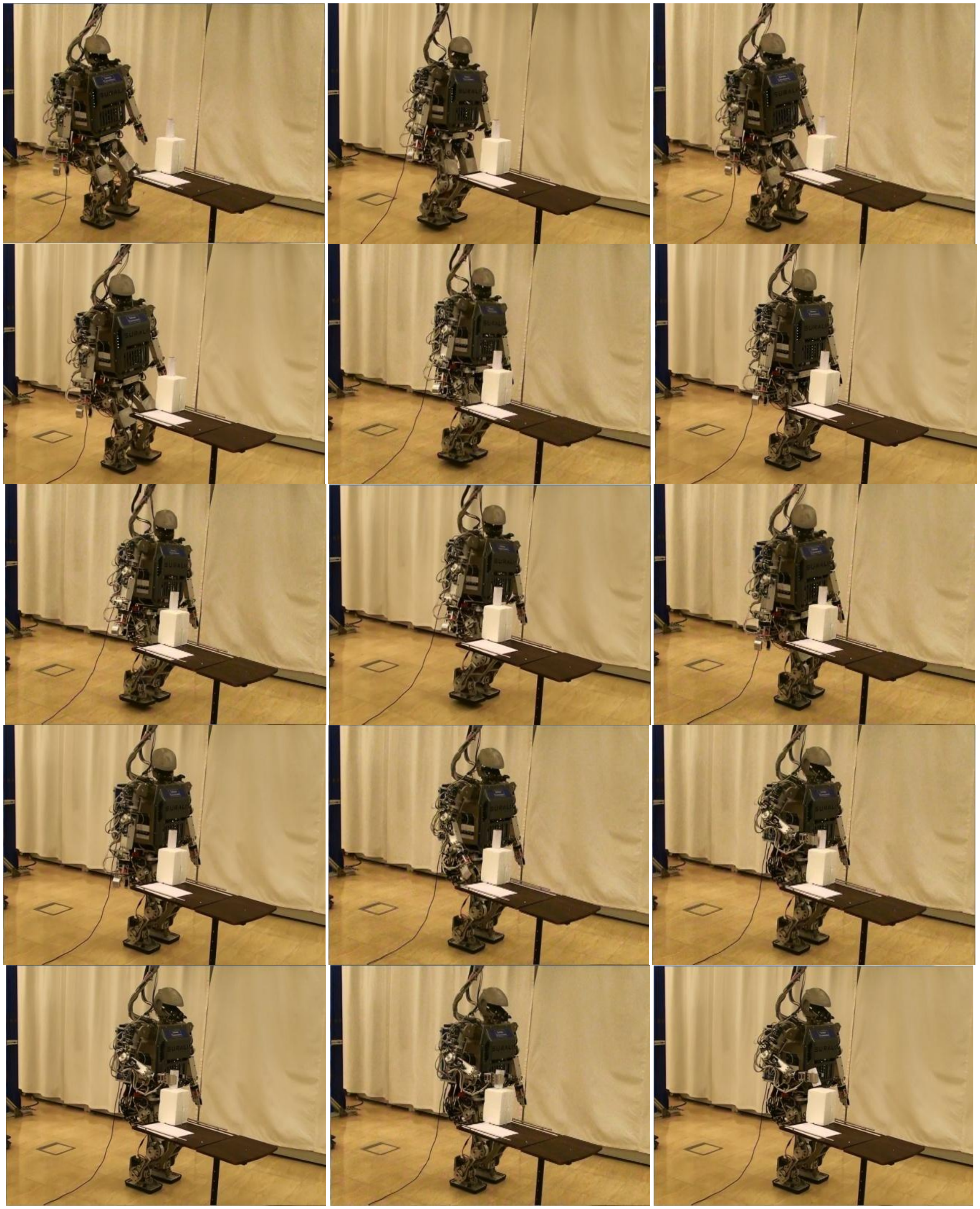


Figure 6.3 The experiment of object grasping by SURALP

Chapter 7

7. DISCUSSION

Experiments show that large amount of oscillations are generated when SURALP walks and these oscillations affect the visual system. The Vision computer is not powerful enough for processing the visual information during the oscillations. The images are processed at the rate of below 15 frames per second. Because of this problem SURALP navigates with single steps and waits for stabilization. This problem can be encountered with a more powerful PC, but due to limited resources it couldn’t be applied in this thesis.

Another problem is the quality of the obtained images. Cameras mounted on SURALP can provide images for precise corner detection up to 1.5 meters. This situation limits SURALPs workspace. SURALP cannot walk parallel to its coronal plane therefore to turn right or left, SURALP also needs to walk forward.

Improvements that can be added to this thesis’ work can be categorized as hardware improvements and algorithm improvements. A more powerful pc and an upgrade to current camera can allow SURALP to manipulate objects with continuous navigation. An object recognition and more complicated object detection algorithm can be used to eliminate the marker on the object so that SURALP can work in any environment and any light source. Also, usually outdoor and indoor environments contain obstacles on the path. Obstacle avoidance algorithm can be added to this work to increase variety of environments that robot can perform the task.

Regardless of these problems objects targeted and inside of the workspace of SURALPs are successfully manipulated. If the object is not inside of robots workspace, robot simply walks to the object and stands in front of it.

Robot can navigate through object regardless of if it can take it or not. If the object is not in workspace of SURALP (which is defined by hardware constraints) robot walks in front of object and takes no further action. Experiments show that robot can take the object with sixty percent success rate. Errors are due to noise from vision data and workspace constraints.

List of contributions of this thesis is shown below.

- First high level task is achieved by SURALP.
- Unique framework is employed for SURALP.
- Employed framework paved the way for future studies on SURALP.

Chapter 8

8. CONCLUSION

This thesis presents an object manipulation algorithm for a humanoid robot using monocular vision. The goal for the humanoid robot is to locate the object and navigate to it with visual feedback, and then grab it from the table. This high level task requires the 3D position information of the object relative to the robot's pelvis frame.

Offline camera calibration is done and camera intrinsic parameters along with distortion coefficients are saved for online use. The object is segmented and tracked with a onetime only user input. Segmentation is done in HSV color space by user defined threshold values. A mask is prepared from segmented image with using morphological operators and finding the largest contour. Then corners of the target object are extracted using Harris Corner Detector. The correspondence is done by assigning labels to corners relative to the direction from objects center point and matching those points with 3D model points with the assumption that requires the object to be parallel to the ground. Using camera intrinsic parameters and distortion coefficients, the pose of the object is calculated.

Image Based Visual Servo (IBVS) control law is derived for a 2-DoF neck to obtain gaze control for humanoid robot. The following assumptions are used in derivation control law;

- The depth parameter is constant (Z).
- Camera velocity vector only contains angular velocities in x and y direction ($v_c = (w_x, w_y)$).

The implementation results are satisfactory for gaze control. The humanoid robot can follow the target object with its head during the walk or standing still.

An algorithm for navigation and object manipulation is presented. Navigation is carried out by using circular arc shaped pattern reference generation, with the 3D position information of the object. The object is grabbed from the table. In this process inverse kinematics are calculated with the object's 3D position data given as the manipulator hand position to obtain each joint's position on the right arm.

The list of assumptions is shown below for object manipulation algorithm.

- Object is identified by marker.
- There is not any bigger object with same color as target object in the scene.

- Object is always parallel to the ground.
- Algorithm is sequential, robot takes a step and locates object and it repeats until it can grasp the object.

The experimental results show that SURALP successfully grabs the object from the table if the object is in the workspace of SURALP. Further workspace restriction in addition to typical workspace of a humanoid robot is observed due to hardware components that are used in this thesis. If the object is not in the constraint workspace of the robot, then SURALP navigates to the object and corrects its pose, and then stays still in front of the object. Single steps are used while navigating to the object due to high oscillations that disturb the pose estimation algorithm.

The proposed algorithm is promising for the following future studies or improvements,

- Robust object recognition algorithm development for the humanoid robot to perform object manipulation tasks in both indoor and outdoor environments with a set of different objects.
- Hardware improvements can allow the robot to continuously walk and make this whole process smoother.

The first application for high level task with visual feedback is employed for SURALP in this thesis work. The framework that is created from implementation of this thesis can be used to establish more high level tasks for SURALP in the future.

REFERENCES

[1] K. Hashimoto, T. Kimoto, t. Ebine, H. Kimura, “Manipulator Control With Image-Based Visual Servo”, *Proc. of 1991 IEEE Int. Conf. on Robotics and Automation,* Vol. 3, pp. 2267-2271, 1991.

[2] N. Hollinghurst, R. Cipolla, “Uncalibrated Stereo Hand-Eye Coordination”, *Proc. British Machine Vision Conference BMVC'93,* pp. 389-398, 1993.

[3] B.H. Yoshimi; P.K. Allen, "Visual Control of Grasping and Manipulation Tasks," *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, 1994,* vol., no., pp.575, 582, 2-5 Oct 1994.

[4] R. Horaud, F. Dornaika, B. Espian, “Visually Guided Object Grasping” , *IEEE Transactions on Robotics and Automation,* Vol. 14, No. 4, pp. 525-532, 1998.

[5] D. Kragic, A. Miller, P. Allen, “Realtime Tracking Meets Online Grasp Planning”, *Proc. of IEEE/RSJ International Conference on Robotics and Automation ICRA’01*, Vol. 3, pp. 2460-2465, 2001.

[6] M. Busnel, R. Cammoun, F. Coulon-Lauture, J. Detriche, G. Le Claire, and B. Lesigne, “The Robotized Workstation "MASTER" For Users With Tetraplegia: Description And Evaluation” *Journal of rehabilitation research and development*, vol. 36, no. 3, pp. 217-229, 1999.

[7] R. Soyama, S. Ishii, and A. Fukase, “The Development Of Meal Assistance Robot MySpoon,” *In Proceedings of the 8th International Conference on Rehabilitation Robotics,* pp. 88-91, 2003.

[8] M. Topping, “An Overview Of The Development Of Handy I, A Rehabilitation Robot To Assist The Severely Disabled” *Journal of Intelligent & Robotic Systems*, vol. 34, no. 3, pp. 253-263, 2002.

[9] H. F. M. Van der Loos, J. J. Wagner, N. Smaby, K. Chang, O. Madrigal, L. J. Leifer, and O. Khatib, “ProVAR Assistive Robot System Architecture” *In IEEE International Conference on Robotics and Automation,* vol. 1. IEEE, pp. 741-746, 1999.

[10] K. Nagatani, S. Yuta, “Designing A Behavior Of A Mobile Robot Equipped With A Manipulator To Open And Pass Through A Door” *Robotics and Autonomous System*, Vo1.17, pp. 5364, 1996.

[11] Y. U Su and W. H. Tsai, “Autonomous Land Vehicle Guidance For Navigation In Building Corridors And Elevators By Computer Vision, Radio, And Photoelectric Sensing Techniques”, *Journal of the Chinese Institute of Engineers,* Vol. 17, No. 1, pp. 63-73, 1994.

[12] K.T. Song; T.Z. Wu, “Visual Servo Control Of A Mobile Manipulator Using One-Dimensional Windows” *Proceedings IECON '99 The 25th Annual Conference of the IEEE Industrial Electronics Society*, vol.2, no., pp.686,691 vol.2, 1999.

[13] Casals, R. Villa, and D. Casals, “A Soft Assistant Arm for Tetraplegics,” *In Rehabilitation technology: strategies for the European Union: proceedings of the 1st TIDE Congress*, p. 103, 1993.

[14] K. Kawamura, S. Bagchi, M. Iskarous, and M. Bishay, “Intelligent Robotic Systems In Service Of The Disabled” *Rehabilitation Engineering, IEEE Transactions on*, vol. 3, no. I, pp. 14-21, 2002.

[15] R. M. Alqasemi, E. J. McCaffrey, K. D. Edwards, and R. V. Dubey, “Wheelchair-Mounted Robotic Arms: Analysis, Evaluation And Development”. *Proceedings IEEE/ASME International Conference on Advanced Intelligent Mechatronics IEEE*, pp. 1164-1169, 2005.

[16] I, Volosyak, O. Ivlev, and A. Graser, “Rehabilitation Robot FRIEND IT -the General Concept and Current Implementation” *ICORR 2005. 9th International Conference on Rehabilitation Robotics*, IEEE, pp. 540-544, 2005.

[17] F. Bley, M. Rous, U. Canzler, and K. F. Kraiss, “Supervised Navigation And Manipulation For Impaired Wheelchair Users” *IEEE International Conference on Systems, Man and Cybernetics,* vol. 3, pp. 2790-2796, 2005.

[18] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, , K. Fujimura, “The Intelligent ASIMO: System Overview And Integration” *IEEE/RSJ International Conference on Intelligent Robots and Systems 2002*, vol.3, pp. 2478- 2483, 2002.

[19] M. Hirose, K. Ogawa, “Honda Humanoid Robots Development”, *Philosophical Transactions of the Royal Society*, Series A, 365 (1850), pp. 11-19, 2007.

[20] K. Nishiwaki, T. Sugihara, S. Kagami, F. Kanehiro, M. Inaba, and H. Inoue, “Design And Development Of Research Platform For Perceptionaction Integration In Humanoid

Robots: H6," in *IEEE/RSJ International. Conference on Intelligent Robots and Systems*, pp. 1559–1564, 2000.

[21] Nishiwaki, K., Kuffner, J., Kagami, S., Inaba, M., & Inoue, H. "The Experimental Humanoid Robot H7: A Research Platform for Autonomous Behaviour". *Philosophical Transaction A: Mathematical Physical and Engineering Sciences*, vol. 365, pp. 79–107, 2007.

[22] M. Hirose, K. Ogawa, "Walking Biped Humanoids That Perform Manual Labour", *Philosophical Transactions of the Royal Society*, Series A, 365 (1850), pp. 65-77, 2007.

[23] H. Hirukawa, F. Kanehiro, K. Kaneko, S. Kajita, K. Fujiwara, Y. Kawai, F. Tomita, S. Hirai, K. Tanie, T. Isozumi, K. Akachi, T. Kawasaki, S. Ota, K. Yokoyama, H. Handa, Y. Fukase, J. Maeda, Y. Nakamura, S. Tachi, H. Inoue, "Humanoid Robotics Platforms Developed in HRP" *Robotics and Autonomous Systems*, Volume 48, Issue 4, Pages 165-175, 2004.

[24] N. Kwak, O. Stasse, T. Foissotte, K. Yokoi, "3D Grid And Particle Based SLAM For A Humanoid Robot". *9th IEEE-RAS International Conference on Humanoid Robots*, pp.62-67, 7-10 Dec. 2009.

[25] J.-Y. Oh, J.-H. Kim, "Realization Of Dynamic Walking For The Humanoid Robot Platform KHR-1", *Advanced Robotics*, vol. 18, no. 7, pp. 749-768, 2004.

[26] J. Y. Kim, I. W. Park, J. H. Oh, "Walking Control Algorithm of Biped Humanoid Robot on Uneven and Inclined Floor", *Journal of Intelligent Robotic System Journal of Intelligent and Robotic System* 48:457-484, January 2007.

[27] J. L. I.W. Park, J.Y. Kim and J. Oh, "Mechanical Design of Humanoid Robot Platform khr-3 (kaist humanoid robot-3: Hubo)," in *IEEE/RAS International Conference on Humanoid Robots*, 2005.

[28] O. Lorch, A. Albert, J. Denk, M. Gerecke, R. Cupec, J. Seara, W. Gerth, and G. Schmidt, "Experiments In Vision-Guided Biped Walking," *in IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2484–2490, 2002.

[29] J. Chestnutt, P. Michel, J. Kuffner, and T. Kanade, "Locomotion Among Dynamic Obstacles For The HONDA ASIMO," *in IEEE Int. Conf. on Robotics and Automation*, pp. 2572–2573, 2007.

[30] P. Michel, J. Chestnutt, S. Kagami, K. Nishiwaki, J. Kuffner, and T. Kanade, “Gpu-accelerated Real-Time 3d Tracking For Humanoid Locomotion And Stair Climbing,” *in IEEE Int. Conf. on Robotics and Automation,* pp. 463–469, 2007.

[31] J.-S. Gutmann, M. Fukuchi, and M. Fujita, “3D Perception and Environment Map Generation for Humanoid Robot Navigation,” *International Journal of Robotics Research,* vol. 27, no. 10, pp. 1117–1134, 2008.

[32] M. Fujita, Y. Kuroki, T. Ishida, “Autonomous Behavior Control Architecture of Entertainment Humanoid Robot SDR-4X”, *Proc IEEE/RSJ International Conference on Intelligent Robots and System,* pp 960-967, 2003.

[33] L. Sentis and O. Khatib. “A Whole-Body Control Framework For Humanoids Operating In Human Environments” In *IEEE Int. Conf. on Robotics and Automation (ICRA’06)*, pages 2641–2648, Orlando, USA, May 2006.

[34] N. Sian, K. Yokoi, S. Kajita, F. Kanehiro, and K Tanie. “A Switching Command-Based Whole-Body Operation Method For Humanoid Robots” *IEEE/ASME Transactions on Mechatronics*, 10(5):546–559, 2005.

[35] M. Inaba, T. Igarashi, S. Kagami, H. Inoue, "A 35 DOF Humanoid That Can Coordinate Arms And Legs In Standing Up, Reaching And Grasping An Object," *International Conference on Intelligent Robots and Systems '96, IROS 96, Proceedings of the 1996 IEEE/RSJ*, vol.1, no., pp.29,36 vol.1, 4-8 Nov 1996.

[36] J. Coelho, J. Piater, and R. Grupen, “Developing Haptic And Visual Perceptual Categories For Reaching And Grasping With A Humanoid Robot,” *Robotics and Autonomous Systems*, vol. 37, no. 2-3, pp. 195– 217, November 2001.

[37] G. Taylor and L. Kleeman, “Flexible Self-Calibrated Visual Servoing for a Humanoid Robot” *In Australian Conference on Robotics and Automation (ACRA2001*, pp. 79–84, 2001.

[38] G. Taylor, L. Kleeman, “Hybrid Position-Based Visual Servoing with Online Calibration for a Humanoid Robot” *Proc IEEE/RSJ International Conference on Intelligent Robots and System*, pp 686-691, 2004.

[39] N. Courty, E. Marchand, and B. Arnaldi, “Through-the-eyes control of a virtual humanoid,” *In IEEE Computer Animation 2001*, H.-S. Ko, Ed., Seoul, Korea, pp. 74–83, 2001.

[40] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue. “Motion planning for humanoids robots” In *Proceeding of the 11th International Symposium of Robotics Research, ISRR*, 2003.

[41] K. Nishiwaki, M. Kuga, S. Kagami, M. Inaba, and H. Inoue. “Wholebody cooperative balanced motion generation for reaching” In *2004 4th IEEE/RAS International Conference on Humanoid Robots*, pages 672–689, Los Angeles, USA, November 2004.

[42] M. Hillman, "Rehabilitation Robotics from Past to Present-A Historical Perspective," *Advances in Rehabilitation Robotics*, pp. 25-44, 2004.

[43] N. Mansard, O. Stasse, F. Chaumette, and K. Yokoi, “Visually-guided grasping while walking on a humanoid robot,” *in IEEE Int. onf. On Robotics and Automation*, ICRA’07, (Roma, Italy), pp. 3041–3047, April 2007.

[44] L. Natale, F. Nori, G. Sandini, and G. Metta, “Learning precise 3d reaching in a humanoid robot,” *IEEE 6th International Conference on Development and Learning*, ICDL 2007, pp. 324–329, IEEE, 2007.

[45] Azad, P.; Asfour, T.; Dillmann, R., "Stereo-based 6D object localization for grasping with humanoid robot systems," *IROS 2007. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007.*, vol., no., pp.919, 924, Oct. 29 2007-Nov. 2 2007.

[46] N. Vahrenkamp, S. Wieland, P. Azad, D. Gonzalez, T. Asfour, and R. Dillmann, “Visual servoing for humanoid grasping and manipulation tasks,” 8th IEEERAS International Conference on *Humanoid Robots*, 2008, pp. 406–412, Dec. 2008.

[47] B. Rasolzadeh, M. Bjorkman, K. Huebner, and D. Kragic, "An Active Vision System for Detecting, Fixating and Manipulating Objects in the Real World," *The International Journal of Robotics Research,* vol. 29, no. 2-3, pp. 133-154, 2010.

[48] Y. Pang; Q. Huang; D. Jia; Y. Tian; G. Junyao; W. Zhang, "Object manipulation of a humanoid robot based on visual Servoing,", *IROS 2007. IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol., no., pp.1124, 1129, Oct. 29 2007-Nov. 2 2007.

[49] K. Erbatur, U. Seven, E. Taskiran, O. Koca, M. Unel, G. Kiziltas, A. Sabanovic, and A. Onat, “SURALP: A New Full-Body Humanoid Robot Platform”, *In the Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Missouri, USA, October 2009.

[50] K. Erbatur, A. Okazaki, K. Obiya, T. Takahashi and A. Kawamura, “A Study On The Zero Moment Point Measurement For Biped Walking Robots”, *Proc. 7th International Workshop on Advanced Motion Control,* pp. 431-436, Maribor, Slovenia, 2002.

[51] Y. Choi, B. J. You, and S. R. Oh, “On the Stability of Indirect ZMP Controller for Biped Robot Systems”, *Proceedings of International Conference on Intelligent Robots and Systems*, pp: 1966 - 1971, vol.2, Sendal, Japan, June 2004.

[52] K. Erbatur, O. Kurt, “Natural ZMP Trajectories for Biped Robot Reference Generation”, *IEEE Transactions on Industrial Electronics*, vol. 56, no. 3, pp. 835-845, March 2009.

[53] K. Erbatur, O. Koca, E. Taskiran, M. Yılmaz and U. Seven, "ZMP Based Reference Generation for Biped Walking Robots," *presented in International Conference on Intelligent Control, Robotics, and Automation, ICICRA 2009*, Venice, Italy October 28-30, 2009, *published in World Academy of Science Engineering and Technology*, Vol. 58, pp. 546-553, October 2009.

[54] A. Dasgupta, and Y. Nakamura, “Making Feasible Walking Motion Of Humanoid Robots From Human Motion Capture Data” *Proceedings of the IEEE International Conference on Robotics and Automation*, Detroit, Michigan, May 1999.

[55] C. Zhu, Y. Tomizawa, X. Luo, and A. Kawamura “Biped walking with variable ZMP, frictional constraint, and inverted pendulum model”, *IEEE International Conference on Robotics and Biomimetics*, pp: 425 – 430, Shenyang, China Aug 2004.

[56] E. Taskiran, M. Yilmaz, O. Koca, U. Seven and K. Erbatur, "Trajectory Generation with Natural ZMP References for the Biped Walking Robot SURALP," Proc. IEEE International Conference on Robotics and Automation, ICRA, Alaska, USA, 2010.

[57] Richard Szeliski. “*Computer Vision: Algorithms and Applications* “Springer-Verlag New York, Inc., New York, NY, USA, 2010.

[58] S. Suzuki, K. be, “Topological Structural Analysis Of Digitized Binary Images By Border Following”, *Computer Vision, Graphics, and Image Processing*, Volume 30, Issue 1, Pages 32-46, 1985.

[59] C. Harris and M. Stephens. “A combined corner and edge detection” *In Proceedings of the Fourth Alvey Vision Conference*, pages 147–151, 1988.

[60] H. Moravec, “Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover” *Tech Report CMU-RI-TR-3, Carnegie-Mellon University, Robotics Institute*, September 1980.

[61] E. Trucco and A. Verri, “*Introductory Techniques for 3-D Computer Vision*” Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[62] D. C. Brown. “Decentering Distortion of Lenses”. 32(3):444–462, 1966.

[63] Z. Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.22, no.11, pp.1330, 1334, Nov 2000.

[64] J.Y. Bouguet, “MATLAB calibration tool”. http://www.vision.caltech.edu/bouguetj/calib_doc/

[65] Helder Ara´ujo Rodrigo L. Carceroni Christopher M. Brown. “A Fully Projective Formulation for Lowe’s Tracking Algorithm” pages 227–238, 1998.

[66] D. Grest. “Marker-Free Human Motion Capture in Dynamic Cluttered Environments from a Single View-Point” *PhD thesis, MIP, Uni. Kiel, Kiel*, Germany, 2007.

[67] K. Levenberg, “A Method for the Solution Of Certain Problems In Least Squares” *Quart. Appl. Math*, Vol. 2, pp. 164–168, 1944.

[68] G. Milighetti, L. Vallone, A. De Luca, "Adaptive Predictive Gaze Control Of A Redundant Humanoid Robot Head," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp.3192,3198, 25-30 Sept. 2011.

[69] F. Nori, L. Natale, G. Sandini, G. Metta, “Autonomous learning of 3D reaching in a humanoid robot,” *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007.* pp.1142, 1147, Oct. 29 2007-Nov. 2 2007.

[70] S. Ushida and K. Deguchi, “Vision-based Motion Control of a Biped Robot Using 2 DOF Gaze Control Structure” *Humanoid Robots, Human-like Machines, Matthias Hackel (Ed.)*, 2007.

[71] D.J. Agravante, J. Pagès, F. Chaumette, “Visual servoing for the REEM humanoid robot’s upper body” *In: Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA*, 2013.

[72] J. Hill and W. Park, "Real time control of a robot with a mobile camera," in *The 9th International Symposium on Industrial Robots*, pp. 233-246, 1979.

[73] B. Espiau, F. Chaumette, and P. Rives. “A New Approach to Visual Servoing In Robotics” *IEEE Trans. on Robotics and Automation*, 8(3):313–326, June 1992.

[74] P. I. Corke, “Visual Control of Robot Manipulators - A Review”, *in Hashimoto, K. (Ed.), Visual Servoing, World Scientific Publishing Co. Pte. Ltd*., pp. 1-31, 1993.

[75] S. Hutchinson, G. Hager & P. Corke, “A Tutorial On Visual Servo Control”, IEEE *Transactions on Robotics and Automation*, 12(5), 651–670, 1996.

[76] D. Kragic and H. I. Christensen, “Survey on visual servoing for manipulation”, *Computational Vision and Active Perception Laboratory (CVAP)*, CVAP259, 2002.

[77] E. Chaumette and S.Hutchinson, "Visual servo control, part I: Basic approaches," *IEEE Robotics and Automation Magazine,* vol. 13, no. 4, pp. 82-90, 2006.

[78] E. Chaumette and S. Hutchinson, "Visual servo control, part II: Advanced approaches," *IEEE Robotics and Automation Magazine,* vol. 14, no. 1, pp. 109-118, 2007.

[79] http://world.honda.com/ASIMO/history/

[80] http://en.wikipedia.org/wiki/HSL_and_HSV

[81] http://en.wikipedia.org/wiki/Distortion_(optics)