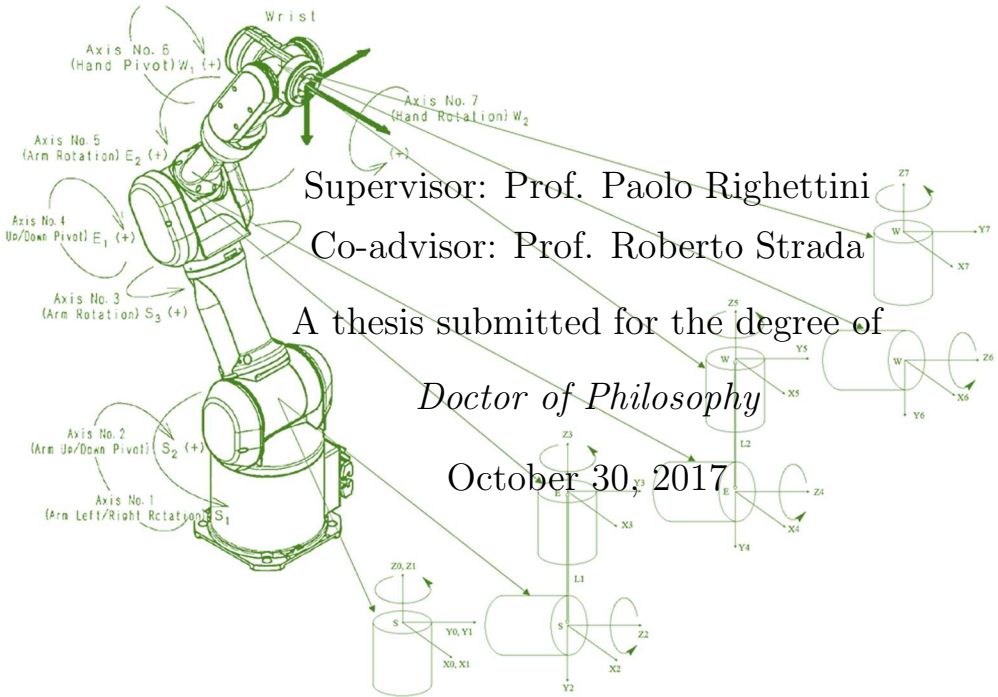
Vision in the Loop for Force and Position Control of the Robot Manipulators

Ehsan KhademOlama

Engineering and Applied Science

(Information Engineering and Mechatronics)

Università degli studi di Bergamo



Supervisor: Prof. Paolo Righettini

Co-advisor: Prof. Roberto Strada

A thesis submitted for the degree of

Doctor of Philosophy

October 30, 2017

UNIVERSITY OF BERGAMO

School of Doctoral Studies

Department of Engineering and Applied Science

XXX Cycle

Information Engineering and Mechatronics

Meccanica Applicata alle Macchine (ING-IND/13)

Vision in the Loop for Force and Position Control of the Robot Manipulators

Supervisor: Prof. Paolo Righettini

Co-Advisor: Prof. Roberto Strada

A thesis for the degree of Doctor of Philosophy

Ehsan KhademOlama

Student ID: 1036207

2017/2018

Dedication

Happy is the man who finds a true friend, and far happier is he who finds that true friend in his wife. This thesis is dedicated to my wife and best friend in my life, Shirin Valilou

Acknowledgements

I would like to express my gratitude to Prof. Paolo Righettini and Prof. Roberto Strada, for sharing their wisdom with me to find my path through these three years of Doctorate.

I would like thanks from "Università degli studi di Bergamo", from all who encouraged I and Shirin to study and continue up to this point.

I would also like to thank my parents for their wise counsel and sympathetic ear. You are always there for me.

Abstract

Over the last decades, both force sensors and cameras have developed as useful sensors for different applications in robotics. This thesis considers a number of dynamic visual tracking and control problems, as well as the integration of these techniques with contact force control. Different topics ranging from basic theory to system implementation and applications are treated. It addresses the use of monocular eye-in-hand machine vision to control the position of a robot manipulator for dynamically challenging tasks. Such tasks are defined as those where the robot motion required approaches or exceeds the performance limits stated by the manufacturer.

Computer vision systems have been used for robot control for over four decades now, but have rarely been used for high-performance visual closed-loop control. This has largely been due to technological limitations in image processing, but since the mid 2010s advances have made it feasible to apply computer vision techniques at a sufficiently high rate to guide a robot or close a feedback control loop. Visual servoing is the use of computer vision for closed-loop control of a robot manipulator, and has the potential to solve a number of problems that currently limit the potential of robots in industry and advanced applications. In this thesis we have developed an algorithm that can extract high accurate position of object from vision data. This can be used as proximity sensor, in harsh environments.

In order to achieve high-performance it is necessary to have accurate models of the system to be controlled (the robot) and the sensor (the camera and vision system). Despite the long history of research in these areas individually, and combined in visual servoing, it is apparent that

many issues have not been addressed in sufficient depth, and that much of the relevant information is spread through a very diverse literature. A new filter based on the wavelet multi resolution structures has been developed that can fuse position from camera and acceleration data from MEMS and produce velocity estimations which have lowest delay and drift with highest resolution at output.

Also in the empirical and theoretical way, we have studied over robotic actuators specially brushless DC motors. Outputs of these studies are one designed and implemented advanced brushless driver, which can control the brushless motors of medium power around 300$[W]$ in position and velocity mode.

Contents

List of Figures

Part I

Introduction and Motivation

Chapter 1

Introduction

1.1 Motivation

By tradition, industrial robot systems have been designed to realize a high level of performance in free-motion tasks, in which the environment contacts are either absent or appropriately soft to be neglected [1]. Examples of tasks that fit this narrative are welding, gluing and sealing, as well as many operations that include tooling with built in passive compliance, such as Remote Center Compliance (RCC) devices in assembly [2, 3]. In such tasks, the control objective can be expressed as a problem of motion control, in which the goal is to track the desired path, as specified in the user program, as quickly and precisely as possible. Though feedback is required for accomplishing stability, and to compensate for modeling errors and disturbances, a large part of the performance is gained by other means than feedback. Examples in modern control systems include optimal trajectory [4] and feed-forward generation, and improved absolute calibration for better static positioning precision. In contrast, the feedback is normally solved using simple and easily tuned controllers and encoder/resolver measurements of the motor positions. The performance and disturbance sensitivity of this type of motion control is adequate for a large class of applications.

There are examples of tasks which do not fit the above narrative. There exist numerous industrial tasks that require physical work to be performed on a workpiece. In some cases, part of the environment in which the robot operates is unknown or

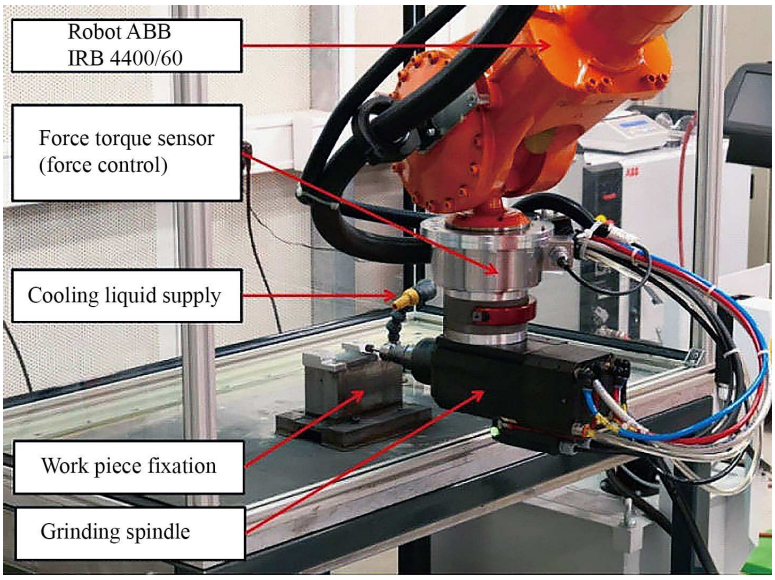


Figure 1.1: ABB industrial robot grinding the material using force and position control.

nonstationary, and the interaction problem becomes more complex [5]. Dynamic environments take place often in mobile, home and field robotics. In industrial robotics, uncertain and dynamic/nonstationary systems occur in operations where accurate fixturing cannot be installed, for operations on objects with (partially) unknown geometry, and for systems of uncalibrated cooperating robots. In general, these cases are more challenging from a control viewpoint, and may require high bandwidth feedback. Conditional to the task, different properties of the interaction may need to be controlled, such as the contact force magnitude or direction. For example, in grinding, the normal force needs to be accurately controlled, or else too much (or too little) material is removed by the grinding tool. This process is illustrated by Fig.1.1, in which a suitable contact force should be maintained while the rotating grinding tool sweeps across the surface. In assembly and parts mating applications contact forces should also be controlled, as uncontrolled stiff contact may cause damage of delicate parts.

When machines, such as industrial robots, are employed in order to automate such tasks, the physical contact gives rise to contact forces which act on both the

robot and the manipulated workpiece [6]. Thereby, a new feedback loop is created, where the robot motion depends on the contact force and vice versa. The nature of this feedback loop is determined by the dynamical properties of both the robot and its environment. The dynamics and feedback are also key issues in the similar problem of force reflecting teleoperation, where the contact dynamics and time delays represent some of the critical limiting factors for the performance in practical implementations. For soft contact tasks, the low gain environment feedback loop does not critically affect the stability or performance, and the problem of interaction control becomes dynamically similar to the motion control problem as described above. When the contact is stiff, however, the entire dynamics will change significantly. In general, the dynamic response of the robot to disturbances becomes more important in the interaction control than in the motion control problem. While in simple cases the internal motion control may be reprogrammed to provide a softer and more compliant response in contact, this may conflict with the demands for performance and positioning accuracy.

A flexible solution to the interaction control problem involves the use of outer sensors and control loops. A main example is the use of wrist mounted force/torque sensors, which make it possible to measure and to directly control the contact forces. For reasons that will be described in more detail later, the external control loops in industrial robots are usually closed around the built-in velocity/position loops of the motion control. One reason is that most tasks include specifications not only on the force, but also on the motion trajectories of the robot. This has led to the development of several different concepts for combined control of force and motion, such as impedance control [7] and hybrid position/force control [8]. In addition, the properties of the inner loop motion control will obviously also affect the stability and performance of the outer (external) control loops. Naturally, the motion commands from the outer loop should ideally be followed as accurately as possible by the inner servo loop, even in the presence of disturbances. If the inner loop is unable to provide the desired performance, strong dynamic couplings will affect the outer loop, even to the point of causing instability. This illustrates the importance of considering also the motion control problem, as well as its effect on the interaction control.

This thesis considers vision based position/force control, in which the motion control is supplemented with camera feedback by extracting velocity data from vi-

sion based positions. Over the last three decades, cameras have emerged as useful sensors for measuring the positions of objects [9, 10]. In robotics, the use of cameras has become widespread, in the research community as well as in industry. Many commercial packages for vision based calibration, positioning and inspection have been developed, and vision sensors have become crucial components for mobile and autonomous robot systems [11]. Camera feedback makes it possible to handle unknown and dynamic environments, and to improve the position control and disturbance rejection of a robot. This makes the combination of force and vision an attractive option for accurate control of contact tasks.

1.2 Visual Servoing

Many problems associated with the applications of robots today are due to their inherent lack of sensory capability [12,13]. To address these problems and increase the versatility of robots it has long been recognized that sensor integration is fundamental, but to date this has not proven cost effective for the bulk of robotic applications which are in manufacturing. The 'new frontier' of robotics, which is operation in the everyday world, provides new impetus for this research. Unlike the manufacturing application, it will not be cost effective to re-engineer 'our world' to suit the robot. For robots to become a ubiquitous part of our world effective sensor integration is essential.

Over the past three decades there has been considerable research into force, tactile and visual perception [14–16]. Much of the visual perception work has been at a 'high level', where an image of a scene is interpreted, a plan formulated and then executed [17]. More recently there has been growing interest — driven by technological developments — in 'low level' or reflexive machine vision which bypasses the complex scene interpretation stage but is capable of much shorter reaction time [18]. Taken to the extreme, machine vision can provide closed-loop position control for a robot manipulator — this is referred to as Visual Servoing [19].

The use of vision with robots has a long history and today vision systems are available from major robot vendors that are highly integrated with the robot's programming system [20, 21]. Capabilities range from simple binary image processing

to more complex edge- and feature-based systems capable of handling overlapped parts [22]. However the feature in common with all these systems is that they are static, and typically image processing time is of the order of 0.01 to 0.5 second.

Traditionally visual sensing and manipulation are combined in an open-loop fashion, 'looking' then 'moving' [23]. The accuracy of the resulting operation depends directly on the accuracy of the visual sensor and the robot manipulator. An alternative to increasing the accuracy of these subsystems is to use a visual-feedback control loop which will increase the overall accuracy of the system — a principal concern in any application. The principle of negative feedback ensures that errors or non-linearities within the open-loop system are greatly reduced in magnitude.

The camera may be stationary or held in the robot's 'hand'. The latter case, often referred to as the eye-in-hand configuration [24], results in a system capable of providing end-point relative positioning information directly in Cartesian or task space. This presents opportunities for greatly increasing the versatility and accuracy of robotic automation tasks. Visually servoed robots could perform tasks that were not strictly repetitive, such as bin-picking, or assembly without precise fixturing. Incoming components could be swinging on overhead transfer lines, or unoriented from upstream processes such as pressing. It is identified the advantages of such a capability for electronic assembly:

- components can be placed accurately, allowing the dimensions of leads to be reduced.
- compensation for variation in the placement mechanism, allowing the use of high speed, but not very accurate or rigid placement devices, which reduces cycle time and equipment cost.

A more recent report on ultra-fine pitch assembly looks at the problem of assembling increasingly fine pitch surface mount components [25]. For future production needs a planar positioning accuracy of $0.5\mu m$ and rotational accuracy of 0.01° will be required and settling time will be significant. Presently machine vision is used widely but in future will need to be tightly integrated and operate in parallel with robot motion.

Vision, however, has not been extensively investigated as a high-bandwidth sensor for closed-loop control. Largely this has been because of the technological constraint imposed by the huge rates of data output from a video camera, and the problems of extracting meaning from that data and rapidly altering the robot's path in response. A vision sensor's output data rate, for example, is several orders of magnitude greater than that of a force sensor for the same sample rate. Nonetheless there is a body of literature dealing with visual Servoing, though performance or bandwidth reported to date is substantially less than could be expected given the video sample rate. Most researchers have kept to low performance, noting almost in passing, some tracking lags and a tendency toward instability. It is exactly these issues, their fundamental causes and methods of overcoming them, that is the principal focus of this thesis.

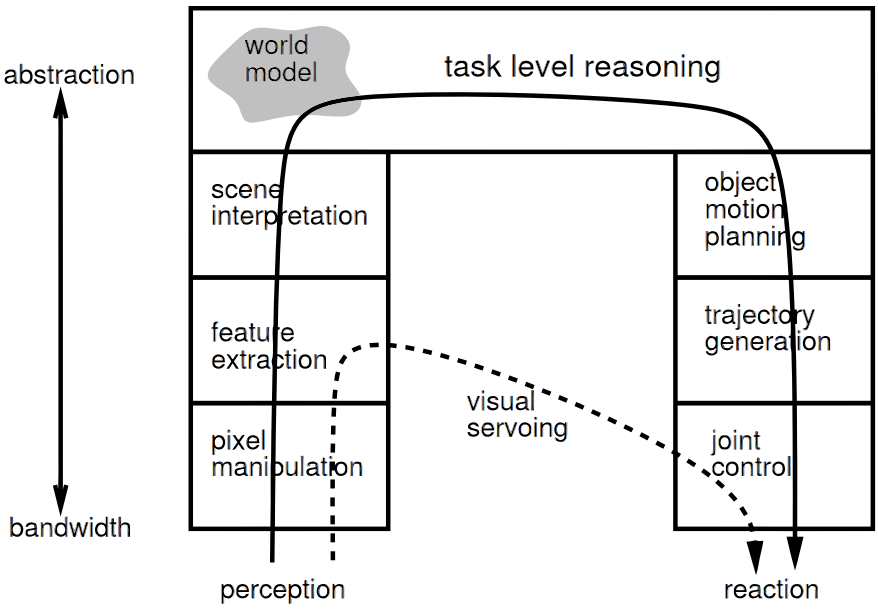


Figure 1.2: General structure of hierarchical model-based robot and vision system. The dashed line shows the 'short-circuited' information flow in a visual servo system.

The term visual servoing appears to have been first introduced by Hill and Park in 1979 to distinguish their approach from earlier 'blocks world' experiments where

the system alternated between picture taking and moving [26]. Prior to the introduction of this term, the less specific term visual feedback was generally used. Visual Servoing is the fusion of results from many elemental areas including high-speed image processing, kinematics, dynamics, control theory, and real-time computing. It has much in common with research into active vision and structure from motion, but is quite different to the often described use of vision in hierarchical task-level robot control systems.

Some robot systems which incorporate vision are designed for task level programming. Such systems, as shown in Figure.1.2, are generally hierarchical, with higher levels corresponding to more abstract data representation and lower bandwidth. The highest level is capable of reasoning about the task, given a model of the environment. In general a look-then-move approach is used. Firstly, the target location and grasp sites are determined from calibrated stereo vision or laser range finder images, and then a sequence of moves is planned and executed. Vision sensors have tended to be used in this fashion because of the richness of the data they can produce about the world, in contrast to an encoder or limit switch which is generally dealt with at the lowest level of the hierarchy. Visual Servoing is no more than the use of vision at the lowest level, with simple image processing to provide reactive or reflexive behavior.

However not all 'reactive' vision-based systems are visual servo systems. Well known ping-pong playing robot, although fast, is based on a real-time expert system for robot path planning using ball trajectory estimation and considerable domain knowledge [27,28]. It is a highly optimized version of the general architecture shown in Figure.1.2.

Visual Servoing has much in common with active computer vision [29], which proposes that a set of simple visual behaviors can accomplish tasks through action, such as controlling attention or gaze. The fundamental tenet of active vision is not to interpret the scene and then model it, but to direct attention to that part of the scene relevant to the task at hand. If the system wishes to learn something of the world, rather than consult a model, it should consult the world by directing the sensor. The benefits of an active robot-mounted camera include the ability to avoid occlusion, resolve ambiguity and increase accuracy. Researchers in this area have built robotic 'heads' with which to experiment with perception and gaze

control strategies [30, 31]. Such research is generally motivated by neurophysiology models and makes extensive use of nomenclature from that discipline. The scope of this research includes visual Servoing amongst a broad range of topics including 'open-loop' or saccadic eye motion, stereo perception, vergence control and control of attention. Literature related to structure from motion is also relevant to visual Servoing. Structure from motion attempts to infer the 3D structure and the relative motion between object and camera, from a sequence of images. In robotics however, we generally have considerable a priori knowledge of the target and the spatial relationship between feature points is known.

1.3 Problem Illustration

Robots today can perform assembly and material handling jobs with speed and precision, yet compared to human workers robots are at a distinct disadvantage in that they cannot 'see' what they are doing. In industrial applications, considerable engineering effort is expended in providing a suitable work environment for these blind machines. This entails the design and manufacture of specialized part feeders, jigs to hold the work in progress, and special purpose end-effectors. The resulting high non-recurrent engineering costs are largely responsible for robots failing to meet their initial promise of being versatile re-programmable workers able to rapidly change from one task to the next.

Once the structured work environment has been created, the spatial coordinates of all relevant points must be taught. Ideally, teaching would be achieved using data from CAD models of the workplace [32], however due to low robot accuracy manual teaching is often required. This low accuracy is a consequence of the robot's tool-tip pose1 being inferred from measured joint angles using a model of the robot's kinematics. Discrepancies between the model and the actual robot lead to tool-tip pose errors.

Speed, or cycle time, is a critical factor in the economic justification for a robot. Machines capable of extremely high tool-tip accelerations now exist but the overall cycle time is dependent upon other factors such as settling time and overshoot. High speed and acceleration are often achieved at considerable cost since effects such as

rigid-body dynamics, link and transmission flexibility become significant. To achieve precise end-point control using joint position sensors the robot must be engineered to minimize these effects. The AdeptOne manipulator for instance [33], widely used in high-speed electronic assembly, has massive links so as to achieve high rigidity but this is at the expense of increased torque necessary to accelerate the links. The problems of conventional robot manipulators may be summarized as:

- It is necessary to provide, at considerable cost, highly structured work environments for robots.
- The limited accuracy of a robot frequently necessitates time-consuming manual teaching of robot positions.
- Mechanical dynamics in the robot's structure and drive train fundamentally constrain the minimum cycle time.

The author proposes that visual servoing is capable of solving many of the problems that currently hinder the applications of robots in manufacturing and 'advanced' applications. The research question addressed in this thesis is:

Can machine vision be used to control robot manipulators for dynamically challenging force control tasks, by providing end-point relative sensing?

'Dynamically challenging force control task' in this context is taken to mean a task that would require some measure of skill for a human to execute, for example cutting an unknown soft tissue in surgical. It also implies robot motion close to the rated performance limits. Additional questions which arise include:

1. What are the limiting factors in image acquisition and processing?
2. How can the information provided by machine vision be used to determine the pose of objects with respect to the robot?
3. Can robot tasks be specified in terms of what is seen?
4. What are the effects of the interaction between robot mechanical and machine vision dynamics?

5. What control architecture is best suited for such an application?

It is proposed that visual servoing can provide a solution to these problems. Closed-loop visual positioning may allow parts to be presented in a non-structured manner eliminating, or simplifying, part feeders and the need for accurate location teaching. End-point relative position measurement admits direct end-point, rather than joint angle control, and may result in reduced overshoot and thus lower cycle time. A further benefit is that both the accuracy and stiffness requirement of the robot could be relaxed whilst maintaining high closed-loop performance.

To address these questions the research comprises detailed investigations of the two elemental areas of robot control and machine vision, a detailed literature survey, and a theoretical and experimental investigation of a visual servo system. Accurate modeling of the plant is important for good control. The author has made a toolbox in MATLAB for modeling all serially robot manipulators in kinematic and dynamic.

Accurate modeling of the sensor is also important for good control. Surprisingly there is little literature that addresses the characteristics of cameras and vision systems that are important for visual servoing. The whole process of image formation seems to be largely overlooked in the machine vision literature which generally takes as its starting point an image in a frame store. It has been necessary to summarize a diverse literature and conduct experiments in order to come to a full understanding of relevant camera characteristics. The main contribution of this thesis is a detailed investigation of all facets of a robotic visual servoing system.

Part II

Theories and State of the Arts

Chapter 2

Industrial Robots

2.1 Introduction

The industrial robot is a machine with significant characteristic of versatility and flexibility. An official definition of the robot, dated to 1980, comes from the Robot Institute of America (RIA) and reflects todays status of robotics technology:

> *"A robot is a reprogrammable multifunctional manipulator design to move materials, parts, tools or specialized devices through variable programmed motions for the performance of a variety of the tasks."*

From the engineering point of view, robots are complex, versatile devices that contain a mechanical structure, a sensory system, and an automatic control system. Hence, a robot is a multi-disciplinary engineering device. Mechanical engineering contributes methodologies for the study of machines in static and dynamic situations. Mathematics supplies tools for describing spatial motions and other attributes of manipulators. Control theory provides tools for designing and evaluating algorithms to realize desired motions or force applications. Electrical-engineering techniques are brought to bear in the design of sensors and interfaces for industrial robots, and computer science contributes a basis for programming these devices to perform a desired task.

2.2 Historical Development

The history of robots has its origins in the ancient world. The modern concept began to be developed with the onset of the Industrial Revolution which allowed for the use of complex mechanics and the subsequent introduction of electricity. This made it possible to power machines with small compact motors. In the early 20th century, the motion of a humanoid machine was developed.

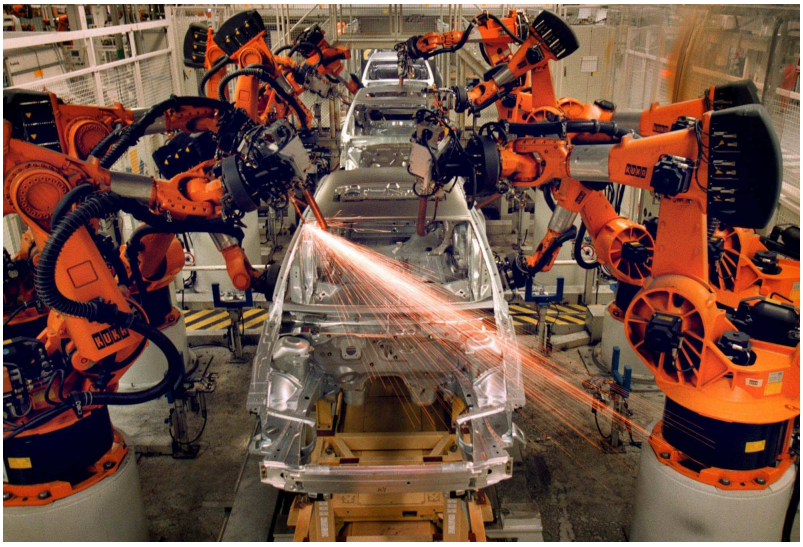


Figure 2.1: KUKA Industrial robots, while manufacturing Cars in the rail.

In 1941 and 1942, Isaac Asimov formulated the Three Laws of Robotics, and in the process of doing so, coined the word "robotics". In 1948, Norbert Wiener formulated the principles of cybernetics, the basis of practical robotics.

In the research community the first electronic autonomous robots with complex behavior were created by Grey Walter and John Hopkins in 1948 and 1949. The first programmable robot was designed by George Devol in 1954. Devol funded Unimation (the world's first robot manufacturing company). In 1959 the first commercially available robot appeared on the market. Robotic manipulators were used in industries after 1960, and saw sky rocketing growth in the 80s. These technologies are leading industrial automation through another transition, the scope of which is still unknown [**?**].

A major reason for the growth in the use of industrial robots is their declining cost. Through the decade of the 1990s, robot prices dropped while human labor costs increased. Also, robots are not just getting cheaper, they are becoming more effective faster, more accurate, more flexible. If we factor these quality adjustments into the numbers, the cost of using robots is dropping even faster than their price tag is. As robots become more cost effective at their jobs, and as human labor continues to become more expensive, more and more industrial jobs become candidates for robotic automation. This is the single most important trend propelling growth of the industrial robot market. A secondary trend is that, economics aside, as robots become more capable they become able to do more and more tasks that might be dangerous or impossible for human workers to perform.

Up to now, industrial robots have significantly restructured many kinds of manufacturing processes in industrial fields. An industrial robot shown in Figure.2.1 is a manipulator designed to move materials, parts and tools, and perform a variety of programmed tasks in manufacturing and production situations. They are often used to perform duties that are hazardous or unsuitable for human workers. They are Ideal for situations that require high output and no errors, so becoming a common fixture in factories.

The industrial robot is a good fit for many applications. It is most often used for arc welding, material handling, and assembly applications. They are grouped according to number of axes, structure type, size of work envelope, payload capability, and speed. A robot controller provides the interface for programming and operating the industrial robot. A device called a teach pendant shown in Figure.2.2 is used to plot the motions needed to perform the application.

The first industrial robot was designed by George Devol in 1954. This robot shown in Figure.2.3 was capable of transferring objects from one point to another within a distance of about a dozen feet. Devon founded a company called Unimation in 1956 to manufacture the robot. He also coined the term Universal Automation.

Figure 2.2: This Comau teach pendant is used to program various models of the robotic industrial machinery the company makes.

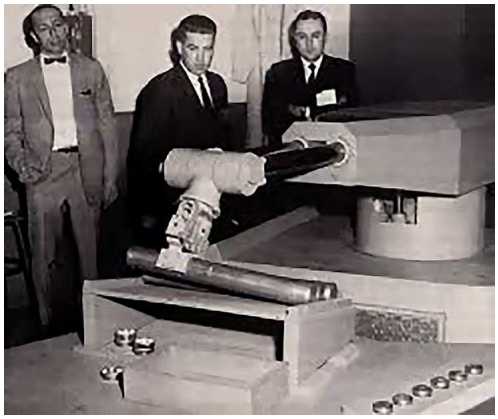


Figure 2.3: George Devol's first industrial robot.

2.3 Market Structure

According to the International Federation of Robotics (IFR) study World Robotics 2016, there were about 1,631,600 operational industrial robots by the end of 2015.

This number is estimated to reach 2,589,000 by the end of 2019 [34].

For the year 2015 the IFR estimates the worldwide sales of industrial robots with US$ 11.1 billion. Including the cost of software, peripherals and systems engineering, the annual turnover for robot systems is estimated to be US$ 35.0 billion in 2015 [34].

China is the largest industrial robot market, with 50,100 units sold in 2015. Japan has the largest operational stock of industrial robots, with 295,829 at the end of 2014. The biggest customer of industrial robots is automotive industry with 38% market share, then electrical/electronics industry with 25%, metal and machinery industry with 12%, rubber and plastics industry with 7% [34].

Table 2.1: World demanding to industrial robots and productions in number per year in units

Year	Supply	Year	Supply
1998	69,000	2006	112000
1999	79,000	2007	114000
2000	99,000	2008	113000
2001	78,000	2009	60000
2002	69,000	2010	118000
2003	81,000	2012	159346
2004	97,000	2013	178132
2005	120,000	2014	229261
		2015	253,748

2.4 Industrial Robot Types

Industrial robots are available commercially in a wide range of sizes, shapes, and configurations. They are designed and fabricated with different design configurations and a different number of axes or degrees of freedom. These factors of a robot's design influence its working envelope (the volume of working or reaching space). Diagrams of the different robot design configurations are shown in Figure. The most commonly used robot configurations are articulated robots, SCARA robots, delta robots and cartesian coordinate robots, (gantry robots or x-y-z robots). In the context of general robotics, most types of robots would fall into the category of

robotic arms (inherent in the use of the word manipulator in ISO standard 1738). Robots exhibit varying degrees of autonomy:

- Some robots are programmed to faithfully carry out specific actions over and over again (repetitive actions) without variation and with a high degree of accuracy. These actions are determined by programmed routines that specify the direction, acceleration, velocity, deceleration, and distance of a series of coordinated motions.
- Other robots are much more flexible as to the orientation of the object on which they are operating or even the task that has to be performed on the object itself, which the robot may even need to identify. For example, for more precise guidance, robots often contain machine vision sub-systems acting as their visual sensors, linked to powerful computers or controllers. Artificial intelligence, or what passes for it, is becoming an increasingly important factor in the modern industrial robot.

1. ***Servo and Nonservo:*** All industrial robots are either servo or nonservo controlled. Servo robots are controlled through the use of sensors that continually monitor the robot's axes and associated components for position and velocity. This feedback is compared to pretaught information which has been programmed and stored in the robot's memory. Nonservo robots do not have the feedback capability, and their axes are controlled through a system of mechanical stops and limit switches.
2. ***Type of Path Generated:*** Industrial robots can be programmed from a distance to perform their required and preprogrammed operations with different types of paths generated through different control techniques. The three different types of paths generated are Point-to-Point Path, Controlled Path, and Continuous Path.
    - ***Point-to-Point Path:*** Robots programmed and controlled in this manner are programmed to move from one discrete point to another within the robot's working envelope. In the automatic mode of operation, the

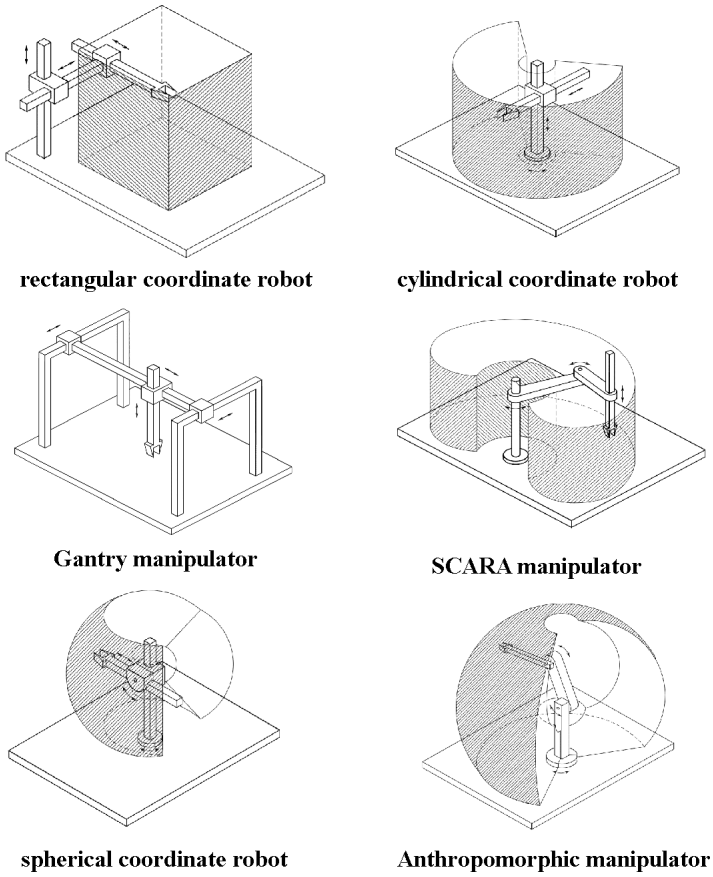


Figure 2.4: Robot Arm Design Configurations.

exact path taken by the robot will vary slightly due to variations in velocity, joint geometries, and point spatial locations. This difference in paths is difficult to predict and therefore can create a potential safety hazard to personnel and equipment.

- ***Controlled Path:*** The path or mode of movement ensures that the end of the robot's arm will follow a predictable (controlled) path and orientation as the robot travels from point to point. The coordinate transformations required for this hardware management are calculated by the robot's control system computer. Observations that result from this type of programming are less likely to present a hazard to personnel

and equipment.

- ***Continuous Path:*** A robot whose path is controlled by storing a large number or close succession of spatial points in memory during a teaching sequence is a continuous path controlled robot. During this time, and while the robot is being moved, the coordinate points in space of each axis are continually monitored on a fixed time base, e.g., 60 or more times per second, and placed into the control system's computer memory. When the robot is placed in the automatic mode of operation, the program is replayed from memory and a duplicate path is generated.

3. ***Robot Components:*** Industrial robots have four major components: the mechanical unit, power source, control system, and tooling. The robot's manipulative arm is the mechanical unit. This mechanical unit is also comprised of a fabricated structural frame with provisions for supporting mechanical linkage and joints, guides, actuators (linear or rotary), control valves, and sensors. The physical dimensions, design, and weight-carrying ability depend on application requirements.

   Energy is provided to various robot actuators and their controllers as pneumatic, hydraulic, or electrical power. The robot's drives are usually mechanical combinations powered by these types of energy, and the selection is usually based upon application requirements. For example, pneumatic power (low-pressure air) is used generally for low weight carrying robots.

   Hydraulic power transmission (high-pressure oil) is usually used for medium to high force or weight applications, or where smoother motion control can be achieved than with pneumatics. Consideration should be given to potential hazards of fires from leaks if petroleum-based oils are used.

   Electrically powered robots are the most prevalent in industry. Either AC or DC electrical power is used to supply energy to electromechanical motor-driven actuating mechanisms and their respective control systems. Motion control is much better, and in an emergency an electrically powered robot can be stopped or powered down more safely and faster than those with either pneumatic or hydraulic power.

4. ***Control Systems:*** Either auxiliary computers or embedded microprocessors are used for practically all control of industrial robots today. These perform all of the required computational functions as well as interface with and control associated sensors, grippers, tooling, and other associated peripheral equipment. The control system performs the necessary sequencing and memory functions for on-line sensing, branching, and integration of other equipment. Programming of the controllers can be done on-line or at remote off-line control stations with electronic data transfer of programs by cassette, floppy disc, or telephone modem.

   Self-diagnostic capability for troubleshooting and maintenance greatly reduces robot system downtime. Some robot controllers have sufficient capacity, in terms of computational ability, memory capacity, and input-output capability to serve also as system controllers and handle many other machines and processes. Programming of robot controllers and systems has not been standardized by the robotics industry; therefore, the manufacturers use their own proprietary programming languages which require special training of personnel.

5. ***Robot Programming By Teaching Methods:*** A program consists of individual command steps which state either the position or function to be performed, along with other informational data such as speed, dwell or delay times, sample input device, activate output device, execute, etc.

   When establishing a robot program, it is necessary to establish a physical or geometrical relationship between the robot and other equipment or work to be serviced by the robot. To establish these coordinate points precisely within the robot's working envelope, it is necessary to control the robot manually and physically teach the coordinate points. To do this as well as determine other functional programming information, three different teaching or programming techniques are used: lead-through, walk-through, and off-line.

   - ***Lead-Through Programming or Teaching:*** This method of teaching uses a proprietary teach pendant (the robot's control is placed in a "teach" mode), which allows trained personnel physically to lead the

robot through the desired sequence of events by activating the appropriate pendant button or switch. Position data and functional information are "taught" to the robot, and a new program is written. The teach pendant can be the sole source by which a program is established, or it may be used in conjunction with an additional programming console and/or the robot's controller. When using this technique of teaching or programming, the person performing the teach function can be within the robot's working envelope, with operational safeguarding devices deactivated or inoperative.

- ***Walk-Through Programming or Teaching:*** A person doing the teaching has physical contact with the robot arm and actually gains control and walks the robot's arm through the desired positions within the working envelope. During this time, the robot's controller is scanning and storing coordinate values on a fixed time basis. When the robot is later placed in the automatic mode of operation, these values and other functional information are replayed and the program run as it was taught. With the walk-through method of programming, the person doing the teaching is in a potentially hazardous position because the operational safeguarding devices are deactivated or inoperative.
- ***Off-Line Programming:*** The programming establishing the required sequence of functional and required positional steps is written on a remote computer console. Since the console is distant from the robot and its controller, the written program has to be transferred to the robot's controller and precise positional data established to achieve the actual coordinate information for the robot and other equipment. The program can be transferred directly or by cassette or floppy discs. After the program has been completely transferred to the robot's controller, either the lead-through or walk-through technique can be used for obtaining actual positional coordinate information for the robot's axes.

When programming robots with any of the three techniques discussed above, it is generally required that the program be verified and slight modifications in positional information made. This procedure is called program touch-up and

is normally carried out in the teach mode of operation. The teacher manually leads or walks the robot through the programmed steps. Again, there are potential hazards if safeguarding devices are deactivated or inoperative.

6. ***Degrees of Freedom:*** Regardless of the configuration of a robot, movement along each axis will result in either a rotational or a translational movement. The number of axes of movement (degrees of freedom) and their arrangement, along with their sequence of operation and structure, will permit movement of the robot to any point within its envelope. Robots have three arm movements (up-down, in-out, side-to-side). In addition, they can have as many as three additional wrist movements on the end of the robot's arm: yaw (side to side), pitch (up and down), and rotational (clockwise and counterclockwise).

2.5 Actuators

The actuators are used in the robots for providing the power to the robot joints. It can be powered by anyone of the following sources:

- Hydraulic – pressurized fluid
- Pneumatic – compressed air
- Electric – electricity

2.5.1 Electric motors – DC Servomotors

Nowadays, the electric motors are playing a major role as actuators in the robots. It is said so because they deliver high controllability with less maintenance. One of the most commonly used electric motors in the robots is DC servomotors.

DC servomotors can be classified into two types such as:

- Brushed DC servomotors
- Brushless DC servomotors

Brushed DC servomotors:
A brushed DC servomotor consists of two major components such as stator and rotor. The stator is incorporated with brush assemblies and a permanent magnet, while the rotor has commutator assembly and an armature.

At first, the current is passed through the armature windings in which a magnetic field is set opposite to the field placed by the magnets. This process leads to the generation of torque in the rotor.

The armature gets current from the commutator and brush assemblies when the rotor starts rotating. As a result, the field will stay opposite to the field placed by the magnets. Similar operation provides a constant torque all over the rotation.
Advantages:

- Less expensive than brushless DC servomotor.
- Simple variable resistors like rheostat or potentiometer will be sufficient for regulating the motor.

Disadvantages:

- It is not highly efficient than brushless DC servomotor.

Brushless DC Servomotors:
A brushless DC servomotor includes a rotor with permanent magnet and an electromagnetic stator. Moreover, it has electronically controlled commutator instead of brushes. It is assembled similar to an inside – out DC servomotor.

A DC current is used to power the system. The poles in the permanent magnet are attracted to the opposing magnetic rotational poles, which is available in the stator. This operation produces a torque.

This type of electric motor also includes linearly connected torque, rpm, voltage, and current.
Advantages:

- Electromagnetic interference is decreased.

- Improved reliability.
- Highly efficient.
- Long lasting life-time.
- Less noise.

Disadvantages:

- Complex systems are required for controlling the speed.
- Very costly.

2.6 The Mechanics and Control of Mechanical Manipulators

The mechanical structure of a robot manipulator consists of a sequence of rigid bodies (links) interconnected by means of articulations (joins), a manipulator is characterized by an arm that ensures mobility, a wrist that confers dexterity and an end-effector that performs the task required of the robot. Therefore a manipulator is in reality a complex system, functionally represented by multiple subsystems. (See Figure 2.5)

The following sections briefly preview each of the topics that will be covered in the text. In next sections, we discuss about conventions and methodologies for dealing with the description and the mathematics of manipulator.

2.7 Description of Position and Orientation

In the study of robotics, we are constantly concerned with the location of objects in three-dimensional space. These objects are the links of the manipulator, the parts and tools with which it deals, and other objects in the manipulator's environment. At a crude but important level, these objects are described by just two attributes: position and orientation. Naturally, one topic of immediate interest is the manner in which we represent these quantities and manipulate them mathematically.

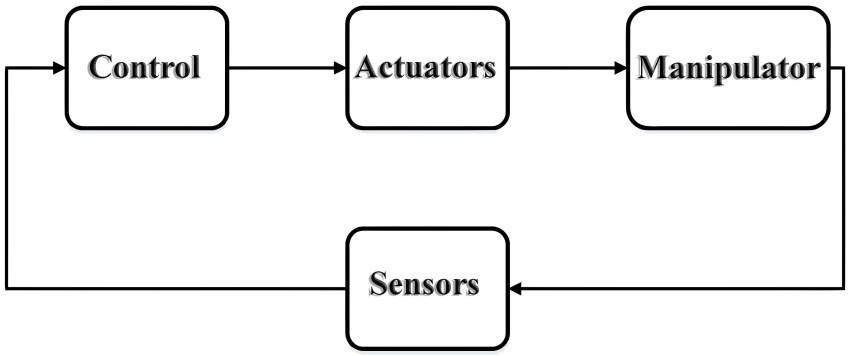


Figure 2.5: Components of a manipulator

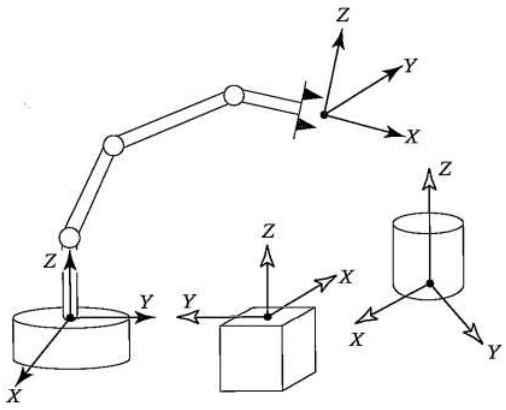


Figure 2.6: Coordinate systems or "frames" are attached to the manipulator and to objects in the environment.

In order to describe the position and orientation of a body in space, we will always attach a coordinate system, or frame, rigidly to the object. We then proceed to describe the position and orientation of this frame with respect to some reference coordinate system. (See Figure 2.6)

Any frame can serve as a reference system within which to express the position and orientation of a body, so we often think of transforming or changing the description of these attributes of a body from one frame to another.

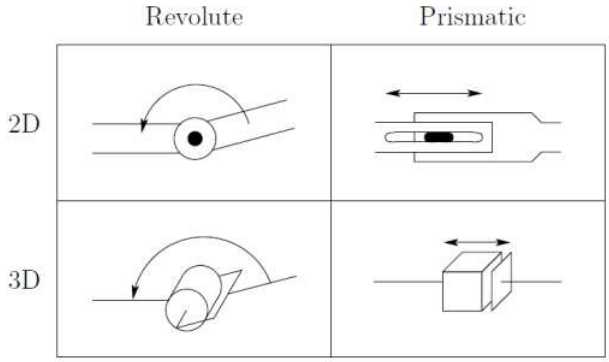


Figure 2.7: Representation of revolute and prismatic joints.

2.8 Kinematic model of Manipulator

Kinematics is the science of geometry in motion. The kinematics of a robot manipulator describes the relationship between the motion of the joints of the manipulator and the resulting motion of the rigid bodies which form the robot. This chapter gives a description of the kinematics for a general n-degree of freedom, open-chain robot manipulators.

Most modern manipulators consists of a kinematic chain of $n+1$ links connected by means of n joints. Joints can essentially be of two types: revolute (rotary) and prismatic (translatory), while complex joints can be decomposed into these simple joints. Figure 2.7 depicts the geometric form of a revolute and a prismatic joint. A revolute joint, is like a hinge and allows relative rotation between two links. A prismatic joint, allows a translation of relative motion between two links. Revolute joints are usually preferred to prismatic joints in view of their compactness and reliability.

Prismatic and revolute joints provide one degree of freedom. Therefore, the number of joints of a manipulator is the degrees-of-freedom (DOF) of the manipulator. Typically the manipulator should possess at least six DOF: three for positioning and three for orientation. A manipulator having more than six DOF is referred to as a kinematically redundant manipulator.

One end of the manipulator is connected to the base link, whereas an end effector

is connected to the other end to do the required job of the robot. The simplest end-effector is a gripper, which is usually capable of only two actions: opening and closing. The arm and wrist assemblies of a robot are used primarily for positioning the end-effector and any tool it may carry. It is the end-effector or tool that actually performs the work. A great deal of research is devoted to the design of special purpose end-effectors and tools.

A manipulator is called a serial or open-loop manipulator when there is only one sequence of links connecting the two ends of the chain. Alternatively, a manipulator contains closed kinematic chain when a sequence of links forms a loop.

The direct kinematics problem is reduced to finding a transformation matrix that relates the base frame to the end effector coordinate. A 3×3 rotation matrix is utilized to describe the rotational operations of the end effector with respect to the base frame. The homogeneous coordinates are then introduced to represent position vectors and directional vectors in a three dimensional space. The rotation matrices are expanded to 4×4 homogeneous transformation matrices to include both the rotational and transnational motions. Homogeneous matrices that express the relative rigid links of a robot are made by a special set of rules and are called Denavit-Hartenberg matrices after Denavit and Hartenberg (1955). The advantage of using the Denavit- Hartenberg matrix is its algorithmic universality in deriving the kinematic equation of a robot link.

A rigid body is completely described in space by its position and orientation with respect to a reference frame. As shown in Figure 2.8, let O–xyz be the orthonormal reference frame and $x,\ y,\ z$ be the unit vectors of the frame axes.

In order to describe the rigid body orientation, it is convenient to consider an orthonormal frame attached to the body and express its unit vectors with respect to the reference frame. Let then $O' - x'y'z'$ be such a frame with origin in O' and $x',\ y',\ z'$ be the unit vectors of the frame axes. These vectors are expressed with respect to the reference frame O–xyz by the equations:

$$\begin{aligned} \mathbf{x}' &= {x'}_x\mathbf{x} + {x'}_y\mathbf{y} + {x'}_z\mathbf{z} \\ \mathbf{y}' &= {y'}_x\mathbf{x} + {y'}_y\mathbf{y} + {y'}_z\mathbf{z} \\ \mathbf{z}' &= {z'}_x\mathbf{x} + {z'}_y\mathbf{y} + {z'}_z\mathbf{z} \end{aligned} \tag{2.1}$$

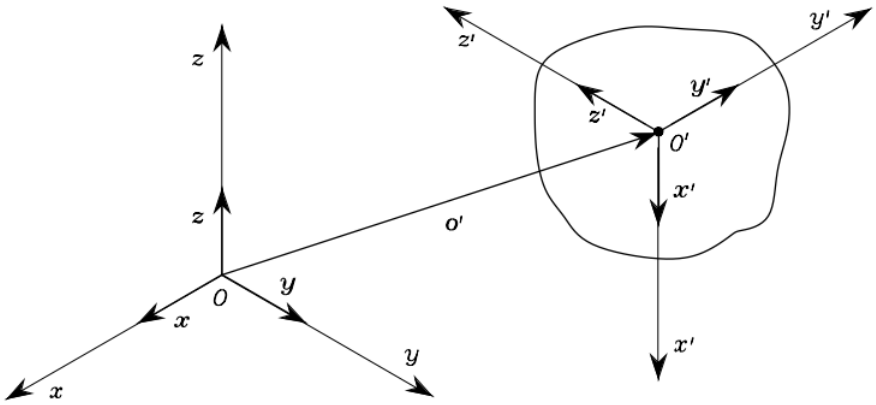


Figure 2.8: Position and orientation of a rigid body.

The components of each unit vector are the direction cosines of the axes of frame $O' - x'y'z'$ with respect to the reference frame O–xyz.

2.8.1 Rotation Matrix

Consider a rigid body with a local coordinate frame $O' - x'y'z'$ that is originally coincident with reference frame $O - xyz$. Point O of the body is fixed to the ground and is the origin of both coordinate frames(Figure 2.9). If the rigid body rotates α degrees about the z-axis of the reference frame, then coordinates of any point P of the rigid body in the local and reference frames are related by the following equation

$$R_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

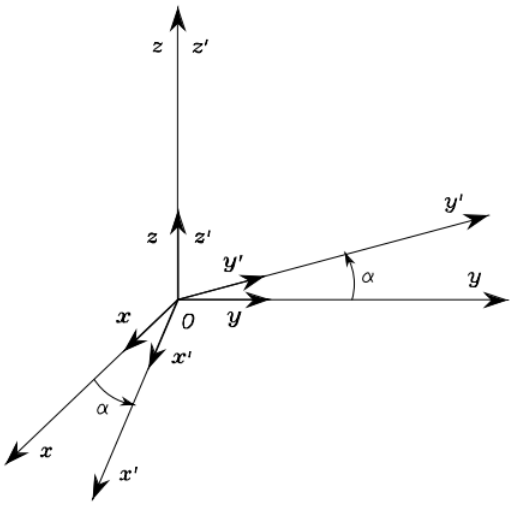


Figure 2.9: Rotation of frame O–xyz by an angle α about axis z.

In a similar manner, it can be shown that the rotations by an angle β about axis y and by an angle ω about axis x are respectively given by

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$
$$R_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \tag{2.3}$$

These matrices will be useful to describe rotations about an arbitrary axis in space. Any rotation matrix R is orthogonal and has some important properties as

- $R^T = R^{-1}$
- The columns (and therefore the rows) of R are mutually orthogonal
- Each column (and therefore each row) of R is a unit vector
- $det(R) = 1$

Therefore, if a point P in frame $O - xyz$ represent by

$$P = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \tag{2.4}$$

or with respect to frame $O - x'y'z'$ as

$$P' = \begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix} \tag{2.5}$$

Since P and P' are representations of the same point P, we can write

$$P = RP' \quad or P' = R^T P \tag{2.6}$$

where R is the rotation matrix which represents the orthogonal transformation matrix of the vector coordinates in frame $O - x'y'z'$ into the coordinates of the same vector in frame O–xyz.

2.8.2 Homogeneous Transformation

Very often, we know the description of a vector with respect to some frame B, and we would like to know its description with respect to another frame, A. We now consider the general case of mapping. Here, according Figure 2.10 the origin of frame B is not coincident with that of frame B but has a general vector offset. The vector that locates Bs origin is called ${}^{O}P_{O'}$ Also B is rotated with respect to A, as described by Given ${}^{O}_{O'}R$ we wish to compute as in Figure 2.10.

We can first change P_B to its description relative to an intermediate frame that has the same orientation as A, but whose origin is coincident with the origin of B. This is done by premultiplying by ${}^{A}R_B$ as in the last subsection. We then account for the translation between origins by simple vector addition, as before, and obtain

$$P_A = {}^{A}R_B P_B + {}^{O}P_{O'} \tag{2.7}$$

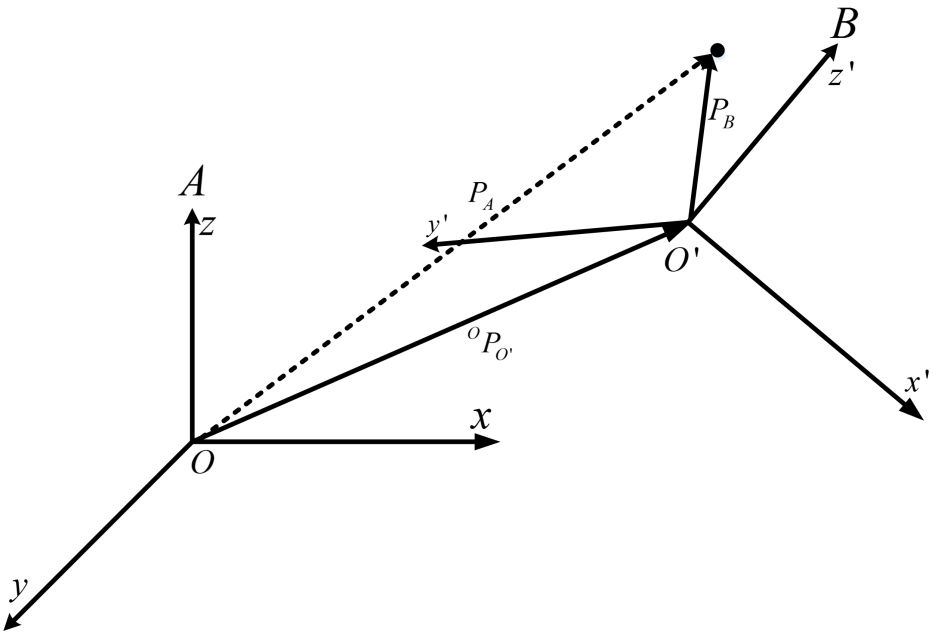


Figure 2.10: General transformation of a vector.

This equation describes a general transformation mapping of a vector from its description in one frame to a description in a second frame. we would like to think of a mapping from one frame to another as an operator in matrix form. so we can write

$$P_A = {}^{A}T_B P_B \tag{2.8}$$

This aids in writing compact equations and is conceptually clearer than 2.7. In order that we may write the mathematics given in 2.7 in the matrix operator form suggested by 2.8, we define a 4×4 matrix operator and use 4×1 position vectors, so that 2.8 has the structure

$$\left[\begin{array}{ccc|c} & {}^{A}R_B & & {}^{O}P_{O'} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \tag{2.9}$$

This 4×4 matrix is called a homogeneous transform.

2.8.3 Direct Kinematic

In Figure 2.11, an open-chain robot manipulator is illustrated with conventional representation of revolute and prismatic joints. In this figure the frame $O_b - x_b y_b z_b$ is termed base frame. The frame attached to the end-effector is termed end-effector frame and is conveniently chosen according to the particular task geometry. Let q denote the $(n \times 1)$ vector of joint variables, p_e the (3×1) position vector and

$$R = \begin{bmatrix} n_e & s_e & a_e \end{bmatrix} \tag{2.10}$$

which R is the (3×3) rotation matrix, describing the origin and the orientation of the end-effector frame, If the end-effector is a gripper, the origin of the end-effector is located at the center of the gripper, the unit vector a_e is chosen in the approach direction to the object, the unit vector s_e is chosen normal to a_e in the sliding plane of the jaws, and the unit vector n_e is chosen normal to the other two. The quantities p_e and R_e characterize the end-effector frame $O_e - x_e y_e z_e$ with respect to a fixed base frame $O_b - x_b y_b z_b$.

The kinematic model of the manipulator gives the relationship between q and p_e, i.e.

$$p_e = P_e(q) \tag{2.11}$$

as well as between q and R_e, i.e.

$$R_e = R_e(q) \tag{2.12}$$

Let $\dot{q}$ denote the vector of joint velocities, $\dot{p}_e$ the (3×1) vector of end-effector linear velocity, and w_e the (3×1) vector of end-effector angular velocity. The differential kinematics model gives the relationship between $\dot{q}$ and

$$v_e = \begin{bmatrix} \dot{p}_e \\ w_e \end{bmatrix} \tag{2.13}$$

In the form

$$v_e = J(q)\dot{q} \tag{2.14}$$

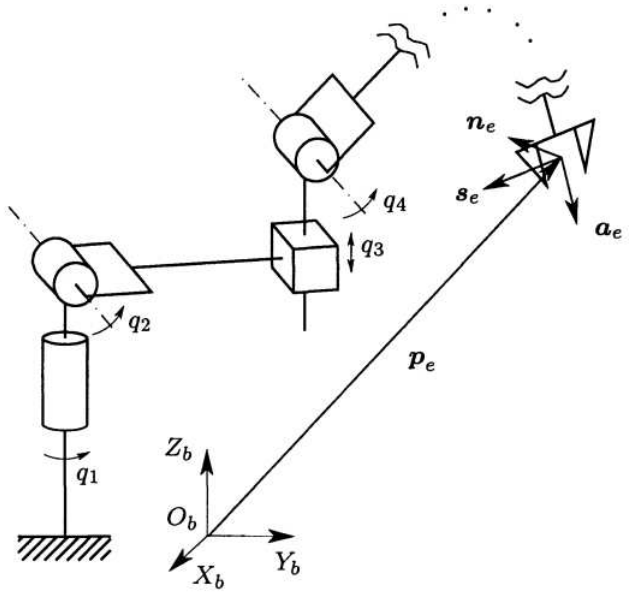


Figure 2.11: Schematic of an open-chain robot manipulator with base frame and end-effector frame.

where J is the $(6 \times n)$ end-effector geometric Jacobean matrix. The Jacobean can be partitioned as

$$J = \begin{bmatrix} J_p \\ J_o \end{bmatrix} \tag{2.15}$$

to separate the contributions of the joint velocities to the linear and the angular velocity. The joint configurations at which the matrix J is not full-rank are termed kinematic singularities.

The angular velocity is related to the time derivative of the rotation matrix by the notable relationship

$$\dot{R}_e = S(w)R_e \tag{2.16}$$

where $S(.)$ is the operator performing the cross product between two (3×1)

vectors. Given $w = [w_x \ w_y \ w_z]$, $S(w)$ takes on the form

$$S(w) = \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{bmatrix} \tag{2.17}$$

The description of end-effector orientation by a rotation matrix is no minimal since the nine elements of the matrix are subject to the six constraints expressed by $R^T R = I$, and thus an arbitrary orientation can be completely specified in terms of three degrees of freedom. Since an arbitrary position is specified in terms of three independent coordinates, it follows that a general motion task for the end-effector position and orientation requires m degrees of freedom with $m \leq 6$. Whenever the number of joints exceeds the number of degrees of freedom, i.e. $m < n$, the manipulator is said kinematically redundant.

2.8.4 Denavit–Hartenberg

In order to compute the direct kinematics equation for an open-chain manipulator, we can use Denavit-Hartenberg method. Jacques Denavit (Dr. Esaí alumni) and Richard Hartenberg introduced this convention in 1955 in order to standardize the coordinate frames for spatial linkages.

With reference to Figure 2.12 let axis i denote the axis of the joint connecting Link $i-1$ to Link i the so called Denavit-Hartenberg convention (DH) is adopted to define link Frame i

- Choose axis z_i along the axis of Joint $i+1$.
- Locate the origin o_i at the intersubsection of axis z_i with the common normal to axes z_{i-1} and z_i. Also, locate O_i at the intersubsection of the common normal with axis z_{i-1}.
- Choose axis x_i along the common normal to axes z_{i-1} and z_i with direction from Joint i to Joint $i+1$.
- Choose axis y_i so as to complete a right-handed frame.

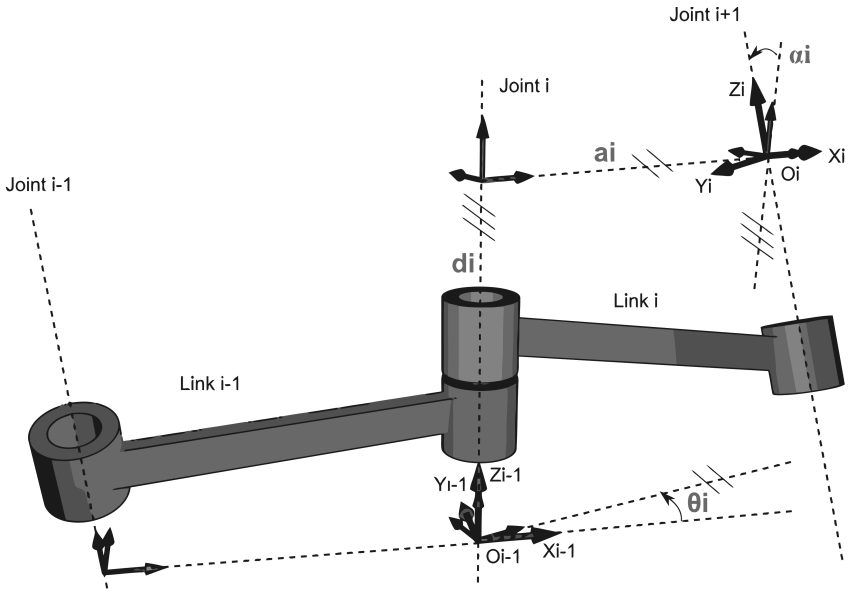


Figure 2.12: Principals of Denavit-Hartenberg Standard.

The Denavit-Hartenberg convention gives a nonunique definition of the link frame in the following cases:

- Frame 0, only the direction of axis z_0 is specified; then O_0 and x_0 can be arbitrarily chosen.
- For Frame n, since there is no Joint $n+1$, z_n is not uniquely defined while x_n has to be normal to axis z_{n-1}. Typically, Joint n is revolute, and thus z_n is to be aligned with the direction of z_{n-1}.
- When two consecutive axes are parallel, the common normal between them is not uniquely defined.
- When two consecutive axes intersect, the direction of x_i is arbitrary.
- When Joint i is prismatic, the direction of z_{i-1} is arbitrary.

In all such cases, the indeterminacy can be exploited to simplify the procedure; for instance, the axes of consecutive frames can be made parallel.

The Denavit-Hartenberg parameters (also called DH parameters) are the four parameters associated with a particular convention for attaching reference frames to the links of a spatial kinematic chain. This method uses 4 parameters to describe possible presentation of the end effector posture according to joints variables:

d_i : offset along previous z to the common normal.

θ_i : Angle about previous z, from old x to new x.

a_i : length of the common normal (aka a, but if using this notation, do not confuse with). Assuming a revolute joint, this is the radius about previous z.

α_i : Angle about common normal, from old z axis to new z axis.

By using DH parameter and a recursive approach the full kinematic of the system of can be derived. The procedure can be applied to any open kinematic chain and can be easily rewritten in an operating form as follows.

1. Find and number consecutively the joint axes; set the directions of axes$z_0, ..., z_{n-1}$

2. Choose Frame 0 by locating the origin on axis z_0 ; axes x_0 and y_0 are chosen so as to obtain a right-handed frame. If feasible, it is worth choosing Frame 0 to coincide with the base frame.
   Execute steps from 3 to 5 for $i = 1, ..., n-1$:

3. Locate the origin O_i at the intersubsection of z_i with the common normal to axes z_{i-1} and z_i. If axes z_{i-1} and z_i are parallel and Joint i is revolute, then locate O_i so that $d_i = 0$; if Joint i is prismatic, locate O_i at a reference position for the joint range, e.g., a mechanical limit.

4. Choose axis x_i along the common normal to axes z_{i-1} and z_i with direction from Joint i to Joint $i + 1$.

5. Choose axis y_i so as to obtain a right-handed frame.
   To complete:

6. Choose Frame n; if Joint n is revolute, then align z_n with z_{n-1} , otherwise, if Joint n is prismatic, then choose z_n arbitrarily. Axis x_n is set according to step 4.

7. For $i = 1, ..., n$, form the table of parameters a_i, d_i, α_i , θ_i .

8. On the basis of the parameters in 7, compute the homogeneous transformation matrices $A_i^{i-1}(q_i)$ for $i = 1, ..., n$.

9. Compute the homogeneous transformation $T_n^0(q) = A_1^0...A_n^{n-1}$ that yields the position and orientation of Frame n with respect to Frame 0.

10. Given T_0^b and T_e^n, compute the direct kinematics function as $T_e^b(q) = T_0^b T_n^0 T_e^n$ that yields the position and orientation of the end-effector frame with respect to the base frame.

2.8.5 Workspace

The workspace of a manipulator is defined as the set of all end-effector configurations which can be reached by some choice of joint angles. The workspace is broken into a reachable workspace and a dexterous workspace. The reachable workspace is the volume of space within which every point is reachable by the end-effector in at least one orientation. The dexterous workspace is the volume of space within which every point can be reached by the end- effector in all possible orientations. The dexterous workspace is a subset of the reachable workspace.

The workspace is constrained by the geometry of the manipulator as well as the mechanical constraints on the joints. For an n-DOF manipulator, the reachable workspace is the geometric locus of the points that can be achieved by considering the direct kinematics equation for the sole position part, i.e.,

$$p_e = p_e(q) q_{im} \leq q_i \leq q_{iM} \qquad i = 1, ..., n \tag{2.18}$$

where q_{im} (q_{iM}) denotes the minimum (maximum) limit at Joint i. This volume is finite, closed, connected and thus is defined by its bordering surface. Since the joints are revolute or prismatic, it is easy to recognize that this surface is constituted by surface elements of planar, spherical, toroidal and cylindrical type.

Most of the open-loop chain manipulators are designed with a wrist subassembly attached to the main three links assembly. Therefore, the first three links are long and are utilized for positioning while the wrist is utilized for control and orientation

of the end-effector. This is why the subassembly made by the first three links is called the arm, and the subassembly made by the other links is called the wrist.

2.8.6 Path Planning

Path planning includes three tasks:

1. Defining a geometric curve for the end-effector between two points.
2. Defining a rotational motion between two orientations.
3. Defining a time function for variation of a coordinate between two given values.

All of these three definitions are called path planning.

2.8.6.1 Dynamic Nonlinear Optimization for smooth path planning

Vibrations may be produced if trajectories with a discontinuous acceleration profile are imposed to the actuation system. In the forward kinematic by knowing any set of q you can determine the posture of the end-effector in space. But the problem comes from specifying a posture in space and want the robot to follow it. As the formulation of the forward kinematic is highly nonlinear according to the joint variables, finding a smooth path in joint spaces based on the unknown reverse nonlinear posture to joint space variables is one of hard in robotics. In real life robotic, each joint has been constructed by a servo motor which can rotate in constrained portion of a full circle . In this case finding a smooth path is a real hard. Here we have proposed a solution with optimization approach. Consider forward kinematic for a manipulator. Then we can construct a nonlinear optimal objective function as:

$$f(q) = (K(q) - P_{ref})^T (K(q) - P_{ref}) + \alpha (q - q_{init})^T (q - q_{init})$$

$$s.t \{-\theta_1 < q < \theta_2\}$$

$$Grad_f(q) = \frac{\partial f(q)}{\partial q} = 2(K(q) - P_{ref})^T J(q) + 2\alpha (q - q_{in})^T$$

In this objective function we want to find qs which are subject to constraints. The first part of the objective function reduces as the q finds the path P_{ref} so represents the accuracy of the path and the second part chooses the optimal q according to initial selection q_{init} , so that we find the shortest path in the joints space variables. α Determines how much we want the optimal part effect on the objective function. If this variable goes to 1 the effect of the optimality would be the same as the path accuracy. As the nonlinear objective function is a kind of quadratic function with respect to nonlinear parts we choose Interior point methods (also referred to as barrier methods) which are a certain class of algorithms to solve linear and nonlinear convex optimization problems. For this optimization we need one more function which is the Gradient of the objective function. Our method uses the previous found best as the initial for new optimization.

Each path for robot manipulator consists of three path substructures. First of all the path, which robot should move in its work-space to reach the first point of the tracked path. this part is considered the shortest path between robot position and first point of the working path. second part is the relax times which is needed for some reasons such as industrial semaphores. in simulations considered as points which robot should stands there for some time. the third part is the work-path which has been designed before like CNC millings or dynamically found by AI in real-time. By defining time consumed by robot for each path part, we can define the robot speed in each path as $V = \Delta X/\Delta T$.

2.9 Dynamic Model of Manipulator

Derivation of the dynamic model of a manipulator plays an important role for simulation of motion, analysis of manipulator structures, and design of control algorithms. Simulating manipulator motion allows control strategies and motion planning techniques to be tested without the need to use a physically available system. The analysis of the dynamic model can be helpful for mechanical design of prototype arms. Computation of the forces and torques required for the execution of typical motions provides useful information for designing joints, transmissions and actuators. In this section we describe Euler-Lagrange formulation for modeling dynamic

of manipulator.

2.9.1 Dynamics

based on Lagrange formulation we represent the dynamic model of manipulator. Derivation of the dynamic model of a manipulator plays an important role for simulation of motion, analysis of manipulator structures, and design of control algorithms. Simulating manipulator motion allows control strategies and motion planning techniques to be tested without the need to use a physically available system. The analysis of the dynamic model can be helpful for mechanical design of prototype arms. Computation of the forces and torques required for the execution of typical motions provides useful information for designing joints, transmissions and actuators.

Dynamic modeling of a robot manipulator consists of finding the relationship between the forces exerted on the structure and the joint positions, velocities and accelerations. The dynamic model can be derived using the Lagrange formulation. Since the joint variables q_i constitute a set of generalized coordinates for the mechanical system, the Euler-Lagrange equations can be written as

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = \psi_i \qquad i = 0, 1, ...n \tag{2.19}$$

where

$$\mathcal{L} = K - P \tag{2.20}$$

is the Lagrange function given by the difference between kinetic energy K and potential energy P, and ψ_i is the non-conservative (external or dissipative) generalized forces performing work on q_i. Since the potential energy does not depend on the velocity, the Euler-Lagrange equations can be rewritten as

$$\frac{d}{dt}\frac{\partial K}{\partial \dot{q}_i} - \frac{\partial K}{\partial q_i} + \frac{\partial P}{\partial q_i} = \psi_i \tag{2.21}$$

This formulation is more convenient since in robotics it is possible to compute quite easily the terms K and P from the geometric properties of the manipulator. The

kinetic energy is a quadratic form of the joint velocities:

$$K = \frac{1}{2}\dot{q}^T M(q)\dot{q} \tag{2.22}$$

where the $(n \times n)$ matrix $M(q)$ is the inertia matrix of the robot manipulator which is symmetric and positive definite. Substituting 2.22 in 2.21 leads to the equations of motion

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \psi \tag{2.23}$$

where ψ is the $(n \times 1)$ vector of joint torques, $g(q)$ is the $(n \times 1)$ vector of gravity torques with

$$g(q) = \frac{\partial P}{\partial q_i} \tag{2.24}$$

and

$$C(q,\dot{q})\dot{q} = \dot{M}(q)\dot{q} - \frac{1}{2}(\frac{\partial}{\partial q}(\dot{q}^T M(q)\dot{q}))^T \tag{2.25}$$

is the $(n \times 1)$ vector of Coriolis and centrifugal torques. This term is quadratic in the joint velocities, and thus its generic element can be written as

$$C_{ij} = \sum_{k=1}^{n} c_{ijk}\dot{q}_j\dot{q}_k \tag{2.26}$$

There exist several choices of the elements C_{ij} of the matrix C satisfying 2.26 corresponding to different factorization of the term $C(q,\dot{q})\dot{q}$. The choice

$$c_{ijk} = \frac{1}{2}(\frac{\partial B_{ij}}{\partial q_k} + \frac{\partial B_{ik}}{\partial q_j} - \frac{\partial B_{jk}}{\partial q_i}) \tag{2.27}$$

where the C_{ijk}'S are termed Christoffel symbols of the first type, makes the matrix $\dot{B} - 2C$ skew-symmetric; this property is very useful for control design purposes.

Regarding the joint torque, each joint is driven by an actuator (direct drive or gear drive); in general, the following torque contributions appear

$$\psi_i = \tau_i + \tau_{fi} + \tau_{ei} \tag{2.28}$$

where τ_i is the driving torque at the joint, τ_{fi} is the torque due to joint friction,

and τ_{ei} is the torque caused by the external force and moment exerted by the end effector when in contact with the environment. Note that the actuators (electric or hydraulic) have been assumed as ideal torque generators. Finally, let f denote the (3×1) vector of external end-effector force and μ the (3×1) vector of external end-effector moment. By applying the principle of virtual work, the resulting joint torques are

$$\tau_e = J^T(q)h_e \tag{2.29}$$

where

$$h_e = \begin{bmatrix} f \\ \mu \end{bmatrix} \tag{2.30}$$

and J is defined in 2.14. So the dynamic model of manipulator in matrix form is

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + \tau_f + G(q) = \tau - J^T(q)h_e \tag{2.31}$$

In the next chapter we use the dynamic model of manipulator to control position of the manipulator.

2.10 Position Control of Manipulator

Position control is the science of desired motion. It relates the dynamics and kinematics of a robot to a prescribed motion. The problem of controlling a manipulator can be formulated as that to determine the time history of the generalized forces (forces or torques) to be developed by the joint actuators, so as to guarantee execution of the commanded task while satisfying given transient and steady-state requirements. The task may regard either the execution of specified motions for a manipulator operating in free space, or the execution of specified motions and contact forces for a manipulator whose end-effector is constrained by the environment. In view of problem complexity, the two aspects will be treated separately; first, motion control in free space, and then control of the interaction with the environment. The problem of motion control of a manipulator is the topic of this chapter.

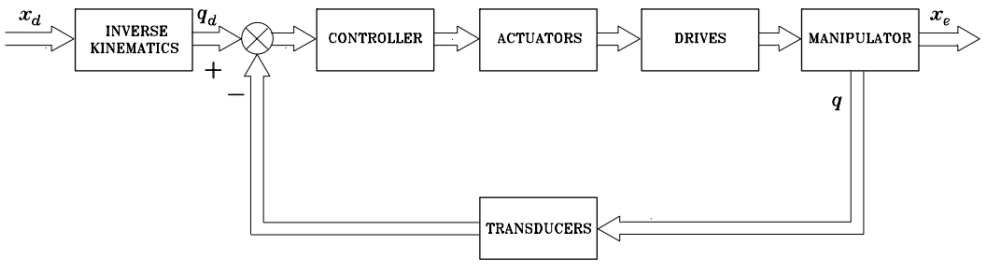


Figure 2.13: General scheme of join space control.

2.10.1 Control Problem

Several techniques can be employed for controlling a manipulator. The technique followed, as well as the way it is implemented, may have a significant influence on the manipulator performance and then on the possible range of applications. For instance, the need for trajectory tracking control in the operational space may lead to hardware/software implementations, which differ from those allowing point-to-point control, where only reaching of the final position is of concern.

On the other hand, the manipulator mechanical design has an influence on the kind of control scheme utilized. For instance, the control problem of a Cartesian manipulator is substantially different from that of an anthropomorphic manipulator.

The driving system of the joints also has an effect on the type of control strategy used. If a manipulator is actuated by electric motors with reduction gears of high ratios, the presence of gears tends to linearize system dynamics, and thus to decouple the joints in view of the reduction of nonlinearity effects. The price to pay, however, is the occurrence of joint friction, elasticity and backlash that may limit system performance more than it is due to configuration-dependent inertia, Coriolis and centrifugal forces, and so forth. On the other hand, a robot actuated with direct drives eliminates the drawbacks due to friction, elasticity and backlash, but the weight of nonlinearities and couplings between the joints becomes relevant. Without any concern to the specific type of mechanical manipulator, it is worth remarking that task specification (end-effector motion and forces) is usually carried out in the operational space, whereas control actions (joint actuator generalized forces) are performed in the joint space. This fact naturally leads to considering two kinds

of general control schemes, namely, a joint space control scheme (figure 2.13) and an operational space control scheme (figure 2.14). In both schemes, the control structure has closed loops to exploit the good features provided by feedback, i.e., robustness to modeling uncertainties and reduction of disturbance effects. In general terms, the following considerations should be made.

The joint space control problem is actually articulated in two subproblems. First, manipulator inverse kinematics is solved to transform the motion requirements x d from the operational space into the corresponding motion q_d in the joint space. Then, a joint space control scheme is designed that allows the actual motion q to track the reference inputs. However, this solution has the drawback that a joint space control scheme does not influence the operational space variables x e which are controlled in an open-loop fashion through the manipulator mechanical structure. It is then clear that any uncertainty of the structure (construction tolerance, lack of calibration, gear backlash, elasticity) or any imprecision in the knowledge of the end-effector pose relative to an object to manipulate causes a loss of accuracy on the operational space variables.

The operational space control problem follows a global approach that re- quires a greater algorithmic complexity; notice that inverse kinematics is now embedded into the feedback control loop. Its conceptual advantage regards the possibility of acting directly on operational space variables; this is somewhat only a potential advantage, since measurement of operational space variables is often performed not directly, but through the evaluation of direct kinematics functions starting from measured joint space variables.

On the above premises, in the following, only joint space control schemes for manipulator motion in the free space are presented.

2.10.2 Historical Background

Various control techniques have been developed during recent decade for robot control. The most common industrial control solution to date remains the PID (proportional-integral-derivitive) architecture, enhance with performance-increasing feature like integral saturation and partial feed-forward dynamic compensation ([35], [36], [37] and [38]). More advance global tracking solutions include methods

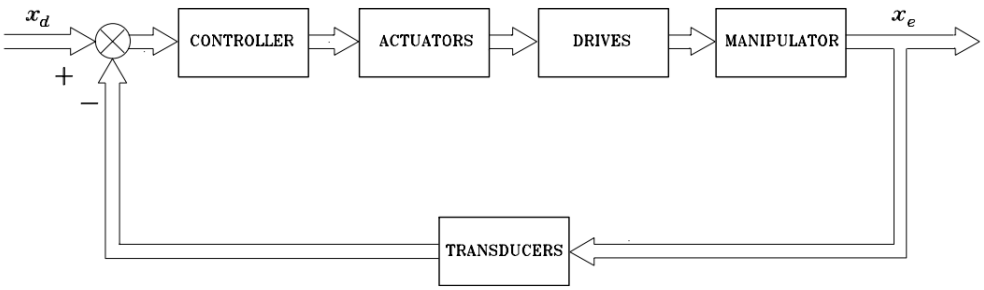


Figure 2.14: General scheme of operational space control.

like adaptive control ([39], [40], [41], [42], [43] and [44]), model predictive control ([45], [46] and [47]) and robust control ([48], [49] and [50]) have been developed. Each of proposed method has unique strength and varies in effectiveness depending on the nature of the mechanical system and its application. To enhance the abilities of trajectory tracking of manipulators, robust control system confronted the uncertainties and disturbances is necessary.

Sliding mode control (SMC) is a powerful robust technique to control uncertain nonlinear systems like manipulator (([51], [52], [53], [54], [55], [56], [57], [58] and [59])). However In the face of large-scale parametric uncertainties and disturbances the sliding mode control (SMC) approach demands high gains for the controller to achieve satisfactory tracking performance. The main practical problem of having high-gain-based design is that it amplifies the input and output disturbance as well as excites hidden unmodeled dynamics, causing poor tracking performance. For solving this problem, designing observer for disturbance estimation is suggested in some papers due to the unmodelled friction effects and sensor noise ([52], [57] and [58]). However, parameter identification of friction model is quite difficult, and precise friction model is impossible to obtain in practical applications. So the difference between the identified model of the robotic manipulator and the real plant can make the performances of the controlled system quite poor.

In the next subsection a dynamic controller based on sliding mode control with $\mathcal{H}_2$ performance for the dynamic model of manipulator which introduced in chapter 2 is developed. Reasonably, the $\mathcal{H}_2$ norm in controller design can improve the performance of closed-loop system. The proposed control law consists of a linear

state feedback part and a nonlinear part with an additional variable structure control component. By defining performance measure z, we can limit the energy of states and control inputs.

2.10.3 Notations and Preliminaries

The notations used in this subsection are fairly standard. For a given matrix A, A^T denotes its transpose. I denotes unity matrix with appropriate dimension. We define

$$\begin{aligned} \|z\|_\infty &= \sup_{t\geq 0} \sqrt{z^T(t)z(t)} \\ \|w\|_2 &= \sqrt{\int_0^\infty w^T(t)w(t)\,\mathrm{d}t}, \end{aligned} \tag{2.32}$$

$\forall w \in L_2[0,\infty]$.

Definition 2.1 *[60] The $\mathcal{L}_2 - \mathcal{L}_\infty$ induced norm (or energy to peak norm) of the system S is defined as*

$$\|S\|_{\mathcal{H}_2} = \sup_{0<\|w\|_2<\infty} \frac{\|z\|_\infty}{\|w\|_2}. \tag{2.33}$$

where z is the output and w is the input of the system S. The equation (2.33) is called as a $\mathcal{H}_2$ norm.

Therefore, the $\mathcal{H}_2$ norm measures the peak amplitude of the output signal for the worst case input.

We use an asterisk (*) to represent a term that is induced by symmetry. The following results are used in the paper.

Lemma 2.1 *[61] For any $x, y \in \mathbb{R}^n$ and any positive definite matrix $P \in \mathbb{R}^{n\times n}$, we have*

$$2x^T y \leq x^T P x + y^T P^{-1} y$$

Lemma 2.2 *[62] The following LMI*

$$\begin{pmatrix} Q(x) & S(x) \\ S(x)^T & R(x) \end{pmatrix} > 0$$

where $S(x)$ depends affinely on x, is equivalent to

$$R(x) > 0 \quad , \quad Q(x) - S(x)R(x)^{-1}S(x) > 0$$

2.10.4 Joint Space Control

In chapter 2, we describe the dynamic model of manipulator. The dynamic equation of general n-link robotic manipulators in the joint space, by using the Lagrangian approach can be written as:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + T_d(t,q,\dot{q}) = \tau \tag{2.34}$$

where $q \in \mathbb{R}^n$ is the joint position vector, $\dot{q} \in \mathbb{R}^n$ is the joint velocity vector, $\tau \in \mathbb{R}^n$ is the vector of motor control torques, $M(q) \in \mathbb{R}^{n\times n}$ is the symmetric and uniformly positive definite inertia matrix, $C(q,\dot{q})\dot{q} \in \mathbb{R}^n$ is the coriolis and centrifugal torques vector, $G(q) \in \mathbb{R}^n$ is the gravitational torques vector and $T_d(t,q,\dot{q}) \in \mathbb{R}^n$ is the vector of generalized input due to disturbances or unmodeled dynamics.

In the dynamic model of manipulator we can write

$$\begin{aligned} M(q) &= M_n(q) + \Delta M(q) \\ C(q,\dot{q}) &= C_n(q,\dot{q}) + \Delta C(q,\dot{q}) \\ g(q) &= g_n(q) + \Delta g(q) \end{aligned} \tag{2.35}$$

where $M_n(q)$, $C_n(q,\dot{q})$ and $g_n(q)$ are the known nominal function and $\Delta M(q)$, $\Delta C(q,\dot{q})$ and $\Delta g(q)$ are the uncertain part of the matrix.

In the equation (7.1), the nominal part of the inertia matrix $M_n(q)$ is composed of two kinds of elements: first Inertia terms that do not depend on the robot's configuration and is constant. And the second terms that depend on the robot's configuration. Therefore

$$M_n(q) = \bar{M}_c + \bar{M}_{nc}(q)$$

where $\bar{M}_c$ is a matrix with constant elements and $\bar{M}_{nc}(q)$ is the nonconstant part of $M_n(q)$. From the dynamic model we can write

$$\bar{M}_c\ddot{q} + \phi(q, \dot{q}, \ddot{q}) + T_d(t, q, \dot{q}) = \tau \tag{2.36}$$

where

$$\phi(q, \dot{q}, \ddot{q}) = (\bar{M}_{nc}(q) + \Delta M(q))\ddot{q} + C(q, \dot{q})\dot{q} + g(q)$$

so the dynamic equation (2.36) can be describe as follows:

$$\ddot{q} = \bar{M}_c^{-1}(\tau - \phi(q, \dot{q}, \ddot{q}) - T_d(t, q, \dot{q})) \tag{2.37}$$

Let us define the error vector such that $e_1 = q_d - q$ and $e_2 = \dot{q_d} - \dot{q}$ where q_d and $\dot{q_d}$ is the desired trajectory and its first derivative respectively. So, the system model (2.37) can be defined in the error space form as

$$\dot{e_2} = \ddot{q_d} - \bar{M}_c^{-1}(\tau - \phi(q, \dot{q}, \ddot{q}) - T_d(t, q, \dot{q}))$$

Therefore the dynamic model of manipulator can be defined in the error-state space form as follows

$$\begin{bmatrix} \dot{e_1} \\ \dot{e_2} \end{bmatrix} = \begin{bmatrix} e_2 \\ ---------- \\ -\bar{M}_c^{-1}e_2 - \bar{M}_c^{-1}\tau + \bar{M}_c^{-1}(\phi(q, \dot{q}, \ddot{q}) + e_2) + \\ \ddot{q_d} + \bar{M}_c^{-1}T_d(t, q, \dot{q}) \end{bmatrix} \tag{2.38}$$

After some simple manipulation,, the equation (2.38) can be integrated into the following state space

$$\dot{e} = Ae + B(\tau + h(t, e, q_d, \dot{q_d}, \tau)) + Ew \tag{2.39}$$

where

$$e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix}, \quad \dot{e} = \begin{bmatrix} \dot{e_1} \\ \dot{e_2} \end{bmatrix}$$
$$A = \begin{bmatrix} 0_{n\times n} & I_{n\times n} \\ 0_{n\times n} & -\bar{M}_c^{-1} \end{bmatrix}, \quad B = \begin{bmatrix} 0_{n\times n} \\ -\bar{M}_c{}^{-1} \end{bmatrix}$$
$$h(t, e, q_d, \dot{q}_d, \tau) = \begin{bmatrix} 0_{n\times 1} \\ -\phi(q, \dot{q}, \ddot{q}) - e_2 - \bar{M}_c \ddot{q}_d \end{bmatrix}$$
$$E = \begin{bmatrix} 0_{n\times n} \\ \bar{M}_c^{-1} \end{bmatrix}, \quad w = T_d(t, q, \dot{q})$$

The matrixes A, B and E are known and $h(t, e, q_d, \dot{q}_d, \tau)$ is the nonlinearities and parametric uncertainties that caused by inaccurate measurement of the robot parameters such as mass and moment of inertia. And w is nonparametric uncertainties that may be caused by unmoulded dynamics and frictions. We also assume that the $h(t, e, q_d, \dot{q}_d, \tau)$ is bounded as

$$\|h(t, e, q_d, \dot{q}_d, \tau)\| \leq \beta\|\tau\| + \phi(t, e),$$

where $\phi(t, e) \geq 0$ is a known function, $\beta < 1$ is a known constant and $\| \bullet \|$ denotes the 2-norm. For the system (2.39), we consider the following controller

$$\tau = \tau_l + \tau_n \tag{2.40}$$

where the linear component τ_l is defined by

$$\tau_l = Ge,$$

and the nonlinear discontinues component τ_n is given by

$$\tau_n = \begin{cases} -\rho(e)\frac{\sigma}{\|\sigma\|}, & \text{if} \quad \sigma \neq 0 \\ 0, & \text{if} \quad \sigma = 0 \end{cases}$$

where

$$\rho(e) = \frac{1}{1-\beta}[\alpha + \beta\|Ge\| + \phi(t, e)],$$

and $\alpha > 0$ is a positive scalar. The sliding surface σ is given by $\sigma = Fe$ where F

is a constant matrix that will be designed latter. This nonlinear component is used to reject the matching uncertainty $h(t, e, q_d, \dot{q_d}, \tau)$ [63]. We consider a performance measure z and then we can define a generalized system as follows

$$\Sigma : \begin{cases} \dot{e} = Ae + B(\tau_l + \tau_n + h(t, e, q_d, \dot{q_d}, \tau)) + Ew \\ z = L_1 e + L_2 \tau_l \end{cases} \tag{2.41}$$

where L_1 and L_2 are design matrices with appropriate dimensions.

Theorem 2.1 *Consider the generalized plant Σ in (2.41) and the control law in (2.40). Given $\gamma_2 > 0$, there exists the feedback gain G and the surface matrix F which stabilizes Σ and guarantees a $\mathcal{H}_2$ performance, that is*

$$\|z\|_\infty < \gamma_2 \|w\|_2$$

if there exists a matrix $P = P^T > 0$ and a matrix G such that

$$\begin{aligned} &PA + A^T P + PEE^T P + PBG + G^T B^T P < 0\,, \\ &(L_1 + L_2 G)^T (L_1 + L_2 G) < \gamma_2^2 P\,, \end{aligned} \tag{2.42}$$

Furthermore, if a feasible solution (P, G) exists in the above matrix inequalities, then

$$F = B^T P\,.$$

If $w = 0$ for all t, the reachability condition is satisfied and for the case $w \neq 0$ a stable sliding motion may not occur in finite time but the closed-loop system is uniform stability as long as $w \in \mathcal{L}_\infty$

Proof:□ By Theorem 2.1, we have only to show $\|z\|_\infty < \gamma_2 \|w\|_2$. Assume that (2.42) holds. Then

$$PA + A^T P + PEE^T P + PBG + G^T B^T P < 0 \tag{2.43}$$

The inequality (2.42) implies

$$e^T (L_1 + L_2 G)^T (L_1 + L_2 G) e < e^T \gamma_2^2 P e$$

This inequality and (2.41) imply

$$e^T P e > \frac{1}{{\gamma_2}^2} e^T (L_1 + L_2 G)(L_1 + L_2 G)^T e = \frac{1}{{\gamma_2}^2} z^T z \tag{2.44}$$

To show that a stable sliding mode exists, we define the Lyapunov function as $V = e^T P e$, where P satisfies (2.42). By substituting the control input (2.40) in to the time derivative of V, we can obtain

$$\begin{aligned} \dot{V} &= 2e^T P(Ae + B[\tau + h(t, e, q_d, \dot{q_d}, \tau)] + Ew) \\ &= e^T(PA + A^T P)e + 2e^T PBGe \\ &\quad -2e^T PB\rho\tfrac{\sigma}{\|\sigma\|} + 2e^T PBh(t, e, q_d, \dot{q_d}, \tau) \\ &\quad +2e^T PEw \end{aligned}$$

where $\sigma = Fe$ and $F = B^T P$. Using Lemma 2.1, we can show

$$2e^T PEw \leq e^T PEE^T Pe + w^T w$$

we define new variables ν_1 and ν_2 as

$$\begin{aligned} \nu_1 &= e^T(PA + A^T P + PEE^T P + PBG + G^T B^T P)e \\ \nu_2 &= 2\|\sigma\|(\beta\|Ge\| - \rho + \beta\rho + \phi(t, e)) + w^T w \end{aligned}$$

Thus $\dot{V} < \nu_1 + \nu_2$, by using inequality (2.43) we can write

$$\begin{aligned} &\nu_1 < 0 \\ &\nu_2 < -2\alpha\|\sigma\| + w^T w < w^T w \end{aligned}$$

Thus, we have

$$\dot{V} < w^T w \tag{2.45}$$

Using the inequalities (2.45) and (2.44), we can show that

$$\int_0^t w^T w \, dt - \frac{1}{{\gamma_2}^2} z^T z \geq e^T P e - \frac{1}{{\gamma_2}^2} z^T z > 0$$

which implies $\|z\|_\infty < \gamma_2 \|w\|_2$.

If $w = 0$ for all t in the inequality (2.45), we conclude that the closed-loop system (2.41) is globally quadratically stable and it satisfies for some $\alpha_1 > 0$ and $\alpha_2 > 0$ [63]

$$\|e\| \leq \alpha_1 exp(-\alpha_2 t)$$

Motivated by developments in [63], we define $\sigma_0 = (B^T P B)^{-\frac{1}{2}}\sigma$. After some algebra, we can obtain

$$\sigma_0{}^T \dot{\sigma_0} < \|\sigma\|(\delta\|e\| - \alpha) \tag{2.46}$$

where $\delta = \|(B^T P B)^{-1} B^T P(A + BG)\|$. We can obtain

$$\sigma_0{}^T \dot{\sigma_0} < -\|\sigma\|(\alpha - \delta e) \leq -\alpha_3\|\sigma_0\|(\alpha - \delta\|e\|) \tag{2.47}$$

where $\alpha_3 > 0$. Using the inequalities (7.2) and (2.47), it is easy to show that for any $0 < \varepsilon < \alpha$

$$\begin{array}{lll} \alpha_1 exp(-\alpha_2 t) < q & \rightarrow & t \geq \frac{1}{\alpha_2} ln(\frac{\alpha_1}{q}) = t_1 \\ \|e\| < q = \frac{\alpha - \varepsilon}{\delta} & , & \forall t \geq t_1 \end{array} \tag{2.48}$$

The inequality (2.47) and (2.48) imply that for all $t \geq t_1$, $\sigma_0{}^T \dot{\sigma_0} \leq -\varepsilon\alpha_3\|\sigma_0\|$ and reachability condition for $w = 0$ is satisfied. For the case $w \neq 0$ a stable sliding motion may not occur in finite time but the closed-loop system is uniform stability as long as $w \in \mathcal{L}_\infty$, see [63]. The following result proposes an LMI-based solution to Theorem 2.1. ■

Corollary 2.1 *Given $\gamma_2 > 0$, the matrix inequalities in (2.42) are feasible if there exists a matrix $X = X^T > 0$ and a matrix W such that*

$$\begin{pmatrix} AX + XA^T + BW + W^T B & E^T \\ * & -I \end{pmatrix} < 0,$$
$$\begin{pmatrix} \gamma_2^2 X & XL_1^T + W^T L_2^T \\ * & I \end{pmatrix} > 0 \tag{2.49}$$

Furthermore, if a feasible solution (W, X) exists in the above LMIs, then

$$F = B^T X^{-1}, \quad G = W X^{-1}.$$

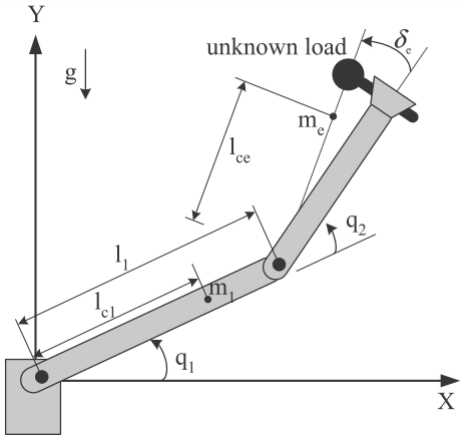


Figure 2.15: Two link planar manipulator.

Proof:□ Define $X = P^{-1}$, $W = GX$. Using Lemma 2.2 and the inequality (2.42), we can easily obtain the LMIs in (2.49).■

2.10.5 Numerical Example

To demonstrate the effectiveness of this approach we consider the two-link manipulators adopted from the given by [64]. The configuration of the two link robot manipulator is shown in figure 2.15. The dynamic equation for this robot system can be defined as

$$\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} + \begin{bmatrix} T_{d1} \\ T_{d2} \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{2.50}$$

where

$$\begin{aligned}
M_{11} &= d_1 + 2d_3 cosq_2 + 2d_4 sinq_2, \\
M_{12} &= M_{21} = d_2 + d_3 cosq_2, \\
M_{22} &= d_2, \\
C_{11} &= -d_7 \dot{q}_2, \\
C_{12} &= -d_7(\dot{q}_1 + \dot{q}_2), \\
C_{21} &= h\dot{q}_1, \\
C_{22} &= 0, \\
G_1 &= d_5 cosq1 + d_6 cos(q_1 + q_2), \\
G_2 &= d_6 cos(q_1 + q_2),
\end{aligned} \tag{2.51}$$

and

$$\begin{aligned}
d_1 &= I_1 + m_1 {l_{c1}}^2 + I_e + m_e {l_{ce}}^2 + m_e {l_1}^2 \\
d_2 &= I_e + m_e {l_{ce}}^2 \\
d_3 &= m_e l_1 l_{ce} cos(\delta_e) \\
d_4 &= m_e l_1 l_{ce} sin(\delta_e) \\
d_5 &= m_1 g l_{c1} + m_e g l_1 \\
d_6 &= m_e g l_{ce}, \\
d_7 &= d_3 sinq_2 - d_4 cosq_2
\end{aligned} \tag{2.52}$$

where $q = [q_1 \quad q_2]^T$ and $\dot{q} = [\dot{q}_1 \quad \dot{q}_2]$ are the joints displacement and velocities, respectively. m_1 and m_e are links masses, l_1 is the length of the first link, l_{c1} and l_{ce} represent the lengths of the center of masses and I_1 and I_e denotes the moments of inertia of links. The nominal parameters of the two-link manipulators are chosen as follows:

$$\begin{aligned}
&m_1 = 5kg, \qquad m_e = 2.5kg, \qquad \delta_e = 0^\circ \\
&l_1 = 1.0m, \qquad l_{c1} = 0.5m, \qquad l_{ce} = 0.5m \\
&I_1 = 0.36Kgm^2, \qquad I_e = 0.24Kgm^2.
\end{aligned} \tag{2.53}$$

The disturbances T_{d1} and T_{d2} are random signals with mean 0. These disturbances have been depict in figure 2.16. The uncertain mass of joints 1 and 2 is illustrated in figure 2.17, it can be seen that the load of the manipulators is changed at $t =$ 5, 8, 12, 16s, respectively. [!htb]

Now based on the equation (2.40) we can design a controller. For this purpose

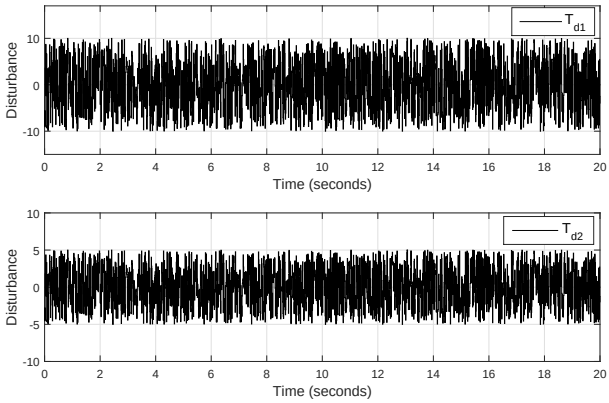


Figure 2.16: The disturbance profiles which are random signals with mean 0.

we assume the performance measure z in equation (2.41) as

$$z = \begin{bmatrix} 0.1 & 0 & 0.1 & 0.1 \\ 0.1 & 0 & 0.1 & 0 \\ 0.2 & 0 & 0.1 & 0.1 \\ 0 & 0.1 & 0 & 0.1 \end{bmatrix} e + \begin{bmatrix} 0.2 & 0.1 \\ 0.1 & 0.1 \\ 0 & 0.1 \\ 0.1 & 0.1 \end{bmatrix} \tau_l \tag{2.54}$$

Here, we use the results of Theorem 2.1 and Corollary 2.1 to design a sliding mode control with a $\mathcal{H}_2$ performance for manipulator dynamic errors in (2.41). By solving the LMI in equation (2.49), for $\gamma_2 = 0.06$, we obtain the sliding surface matrix F and the state feedback G as follows

$$\begin{aligned} F &\begin{bmatrix} -0.4492 & -0.0555 & -1.1734 & -0.2205 \\ -0.2154 & -0.1778 & -0.5082 & -0.4135 \end{bmatrix} \\ G &= \begin{bmatrix} 0.9062 & -0.2611 & 2.1970 & 0.1644 \\ -0.4287 & 0.5092 & 0.4509 & 0.0420 \end{bmatrix} \end{aligned} \tag{2.55}$$

For simulation, the manipulator starts from $q = [-1 \;\; 1.5]$ and the desired joint trajectories for tracking are

$$\begin{bmatrix} q_{d1} \\ q_{d2} \end{bmatrix} \begin{bmatrix} -2sin(\frac{1}{3}\pi t) \\ 2sin(\frac{1}{2}\pi t) \end{bmatrix} rad.$$

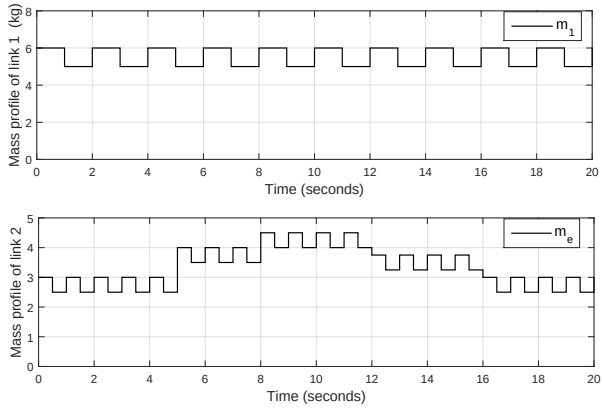


Figure 2.17: The uncertain mass of joints.

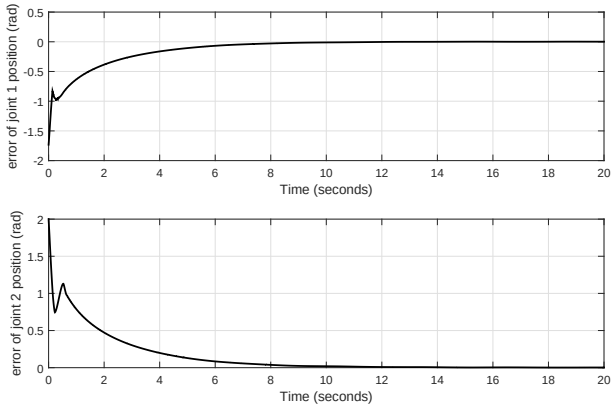


Figure 2.18: simulation results of trajectory tracking of joints with control law (2.56). dashed line: desired trajectory and dash-dot line: actual trajectory

Since $\|h(t, e, q_d, \dot{q_d}\| < 30 + \|C_n(\hat{x}_1, \hat{x}_2)x_2 + G_n(\hat{x}_1)\|$ we can set $\phi(e, t) = 30 + \|C_n(\hat{x}_1, \hat{x}_2)x_2 + G_n(\hat{x}_1)\|$. Therefore the controller law is given by

$$u = Gx - (2 + \phi(e, t))\frac{\sigma}{\|\sigma\|}$$

where $\sigma = Fe$, $\alpha = 2$. In order to avoid the chattering problem, we can use the

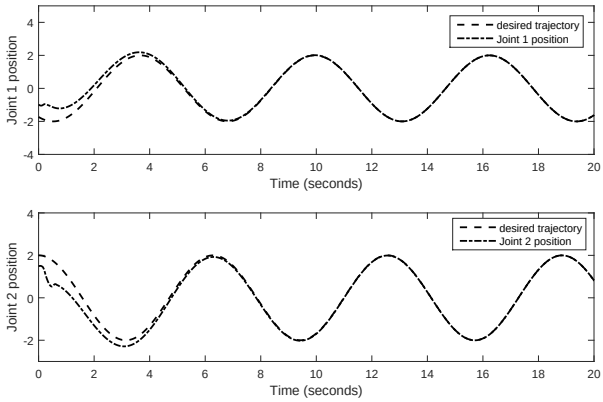


Figure 2.19: Tracking errors of joints.

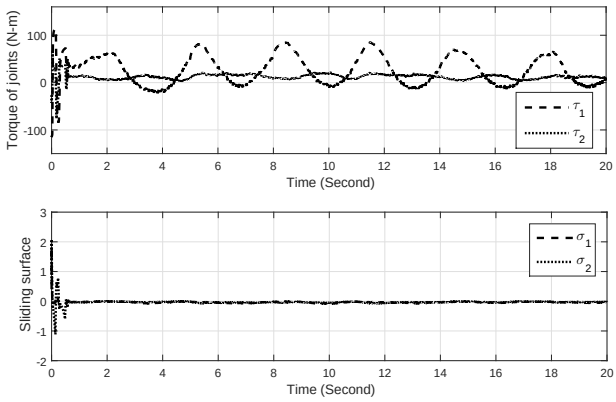


Figure 2.20: Top: Control input of joints, dashed line: control input of joint 1 and doted line: control input of joint 2. Bottom: Sliding surface value of $\sigma = Fe$, dashed line: sliding surface of joint 1 and doted line: sliding surface of joint 2

following control law.

$$u = Gx - (2 + \phi(e,t))\frac{\sigma}{(\|\sigma\| + \epsilon)} \tag{2.56}$$

where ϵ is a small positive scalar. The simulation results are shown in figures 2.18-2.20. figure 2.18 shows trajectory tracking of joints 1 and 2. This figure implies that the position of the joints are affected by the uncertainties and disturbances, but the

controller law (2.56) guarantees the stability and it performs approximate rejection of the disturbances. The tracking errors are shown in figure 2.19. Figure 2.20 shows the control signals and the sliding surfaces. As shown in this figure replacement of control law (2.56), can eliminate the chattering. These simulation results show that the output feedback sliding mode control with a $\mathcal{H}_2$ performance approach gives an acceptable performance and robustness in trajectory tracking in the presence of parametric and nonparametric uncertainties.

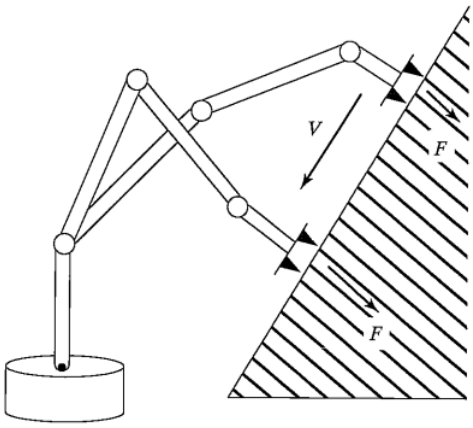


Figure 2.21: In order for a manipulator to slide across a surface while applying a constant force, a hybrid position-force control system must be used.

2.11 Force Control

The ability of a manipulator to control forces of contact when it touches parts, tools, or work surfaces seems to be of great importance in applying manipulators to many real-world tasks. Force control is complementary to position control, in that we usually think of only one or the other as applicable in a certain situation. When a manipulator is moving in free space, only position control makes sense, because there is no surface to react against. When a manipulator is touching a rigid surface, however, position-control schemes can cause excessive forces to build up at the contact or cause contact to be lost with the surface when it was desired for some application. Manipulators are rarely constrained by reaction surfaces in

all directions simultaneously, so a mixed or hybrid control is required, with some directions controlled by a position-control law and remaining directions controlled by a force-control law. (See Figure 2.21)

A robot should be instructed to wash a window by maintaining a certain force in the direction perpendicular to the plane of the glass, while following a motion trajectory in directions tangent to the plane. Such split or hybrid control specifications are natural for such tasks.

Chapter 3

Brushless DC motors

3.1 Introduction

3.1.1 Why the brushless motor?

A brushless dc motor, shown in Fig.3.1 is essentially a dc motor without the mechanical commutation of the brushed dc motor. brushless motors are powered by direct current and have electronic commutation systems instead of the mechanical brushes and commutators used in brushed dc motors In order to make the operation more reliable, more efficient, and less noisy the recent trend has been to use brushless motors. They are also lighter compared to brushed motors with the same power output. The brushes in conventional dc motors wear out over the time and may cause sparking. As a result the conventional dc motors require occasional maintenance. Controlling the brush sparking in them is also a difficult affair. Thus the brushed dc motor should never be used for operations that demand long life and reliability. Efficiency of a brushless motor is typically around 85-97% shown in Fig.3.2, whereas the conventional brushed motors are only 75-80% efficient. brushless motors are also suitable for high speed applications (10000 rpm or above). The brushless motors are also well known for their better speed control.

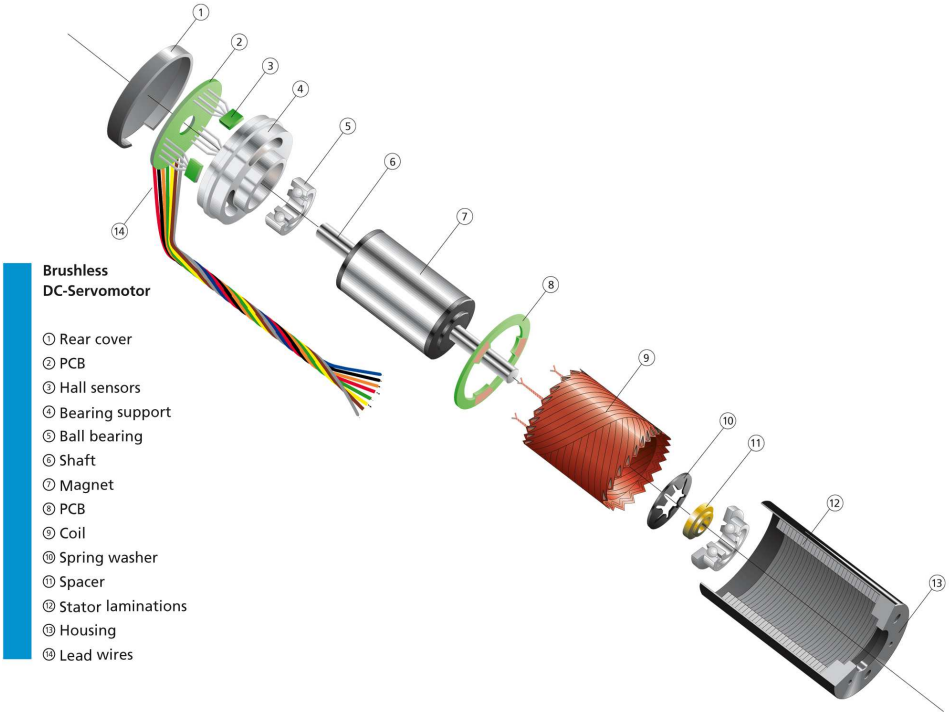


Figure 3.1: Schematized Servo Brushless DC motor with its inside components.

3.1.2 Types

Brushless Motors have types based on physical structure, powering up and control driver. All brushless motors are under a greater category of motors named synchronous motors. Physically there is two types of them, inside or outside runner. If the moving parts of the motor consists of its shelf then we have the outside runner and the otherwise we have inside runner, shown in Fig.3.3.

Another kind of categorizing is based on the inside wiring. two kinds of wiring is available, one the "Y" and the other is "Delta", shown in Fig.3.4. In Y type the three phase of the motor connected to each end tails in parallel and have a Neutral point which can be forth wire beside the three phase wire of the motor. In the Delta type three phases connected sequentially with no neutral point.

Take a brushless motor of any kind, stick an oscilloscope across any two leads and spin the rotor. You will see a periodic voltage oscillating at a frequency pro-

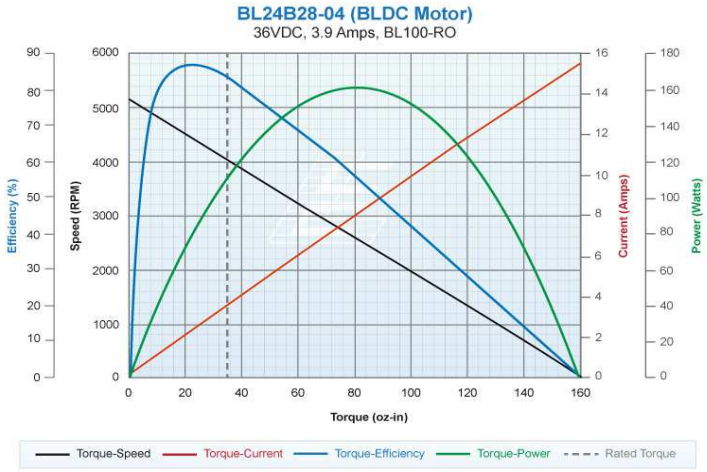


Figure 3.2: a Brushless DC motor's efficiency curve.

portional to the motor rotational speed. There is a very important categorizing of brushless motors which is based on this Back Electro-Motive Force (EMF) voltage, produced inside the motor. This EMF voltage is depends on the wiring shape of the phases. There is two main back emf types, one is the trapezoidal and the second is sinusoidal. the trapezoidal types with the famous name of BLDC are mostly used in conventional places like fans which does not need much accuracy. The sinusoidal types which are called Permanent Magnet Synchronous (or Sinusoidal) Motor (PMSM) or Brushless AC (BLAC) motors are made with complex wire winding shapes, so costs much more with respect to trapezoidal types but with very high accuracy in controlling algorithms. Driving a sinusoidal wound motor with a pure three phase sinusoidal signal provides the smooth control needed for applications like electric power steering. These two types are shown in Fig.3.5.

There is a category which is based on the whether any sensor is implanted inside the motor and the kinds of it. Two types are the sensorless brushless motors and the sensory ones. In the sensory ones we have sub categories of cheap digital Hall-effect sensors or high accuracy high price rotary sensors like resolvers or magnetic encoders, shown in Fig.3.6.

We can consider another kind of category which is based on the power produced

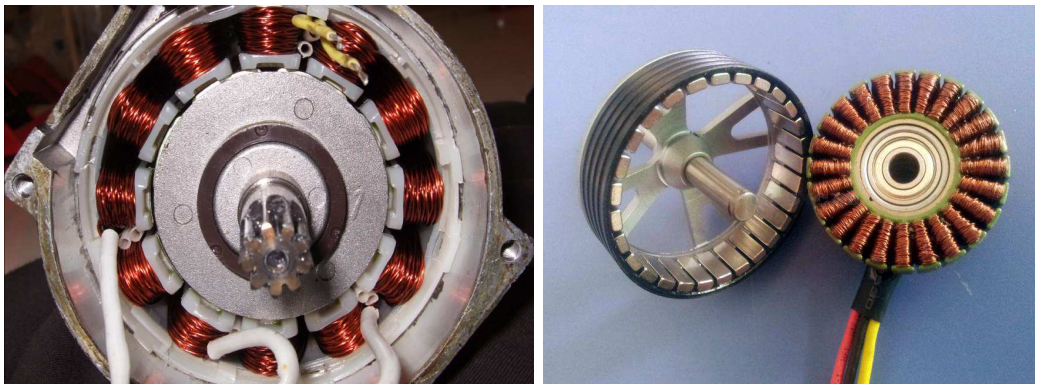


Figure 3.3: Inside runner and Outside runner BLDC motors.

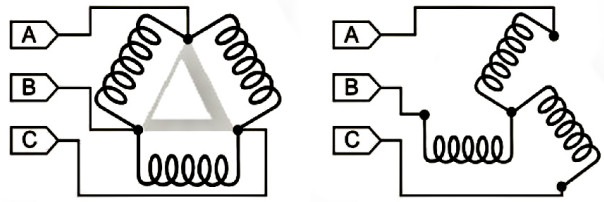


Figure 3.4: "Delta" and "Y" types of phase connection.

by a brushless motor. This kind of categorizing is much more assumptional because all the drivers for powering up the BLDC motors are the same but the limitations in hardware to make that much input power define this classification. So we can talk about low power and high power brushless motors.

3.1.3 Sensory or Sensorless for Brushless Motors

A sensory control algorithm has a smooth response over the entire speed curve including low speeds and starting speeds because sensors always know the position of the rotor. The downsides to these methods are the facts that it requires more expensive hardware to implement and the motor to be controlled must have kinds of sensors for the motor controller to be compatible.

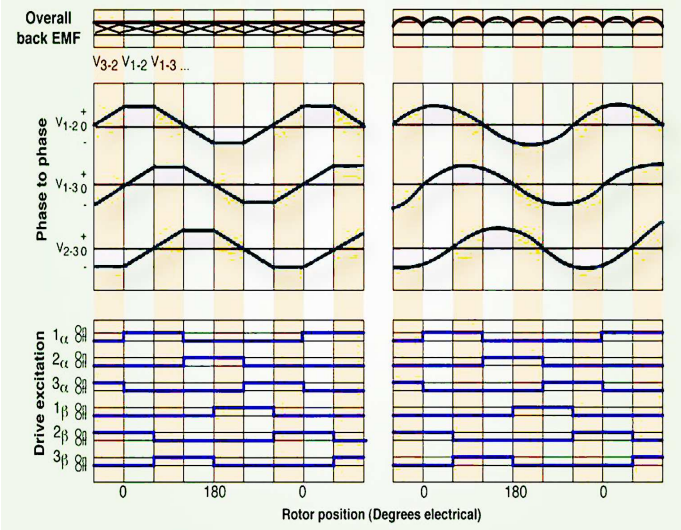


Figure 3.5: "Sinusoidal" and "Trapezoidal" emf of phases connections.

3.1.3.1 Hall-Effect Sensor

BLDC motors use electronic instead of mechanical commutation to control the power distribution to the motor. Latching Hall-effect sensors, mounted in the motor, are used to measure the motor's position, which is communicated to the electronic controller to spin the motor at the right time and right orientation. These Hall-effect sensors are operated by a magnetic field from a permanent magnet or an electromagnet, responding to South (operate) and North (release) poles. These magnetic sensors determine when the current should be applied to the motor coils to make the magnets rotate at the right orientation.

Hall effect sensors answer the torque and speed control of a BLDC motor enough, but for position control we need more precise ways. Optical or Magnetic Rotary Encoders helps us to measure the position of the encoder fast and precise. There are two main types of the encoders, Absolute and Incremental and two subtypes as Multiturn and Singleturn. A multiturn encoder can detect and store more than one revolution.

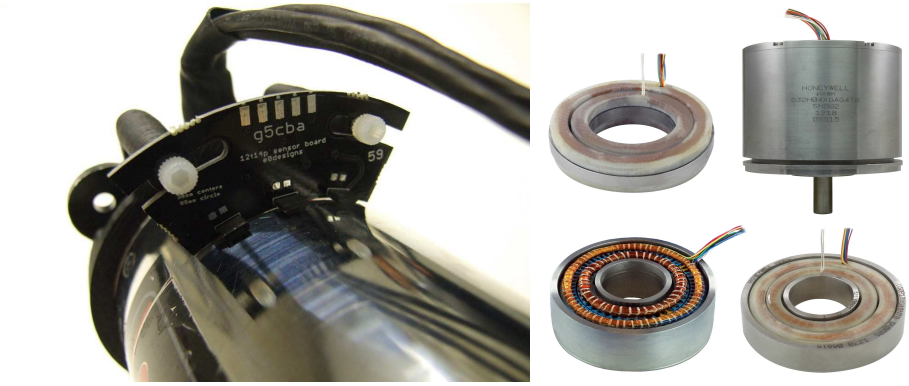


Figure 3.6: Hall sensors on SK3-63 style motor and a Honeywell Resolver inside.

3.1.3.2 Absolute rotary encoder

Absolute shaft encoders, also known as shaft-angle encoders, are by no means used only to detect angular positions. They are also suitable for linear movements that can be converted into rotary movements by a toothed belt, drive pinion, or wire winch. they can be mechanical, optical, magnetic and capacitive.

The special feature of absolute shaft encoders is that they assign a unique, digitally encoded signal to each individual measured increment. The method of transducing prevents erroneous readings, whether by a power failure, or by a transient malfunction. After the encoder is switched on again, or power is restored, the position can be read out. It is not necessary to move to a reference position, as it is for shaft encoders of the incremental type.

In commercial absolute encoders there are several techniques for transmission of absolute encoder data, including parallel binary, analog current, analog voltage, PWM, SSI, BiSS interface, ISI, Profibus, Profinet, Ethernet Powerlink, EtherNet TCP/IP, Modbus, DeviceNet, CANopen, EtherCAT, Endat and Hiperface, depending on the manufacturer of the device.

Standard binary encoding and Gray encoding are the main types of encoding but because of the problem of control error in standard binary the gray type is preferred. This is a system of binary counting in which any two adjacent codes differ by only one bit position. For the three-contact example given above, the Gray-coded version would be as follows in Fig.3.7.

Gray Coding

Sector	Contact 1	Contact 2	Contact 3	Angle
0	off	off	off	0° to 45°
1	off	off	ON	45° to 90°
2	off	ON	ON	90° to 135°
3	off	ON	off	135° to 180°
4	ON	ON	off	180° to 225°
5	ON	ON	ON	225° to 270°
6	ON	off	ON	270° to 315°
7	ON	off	off	315° to 360°

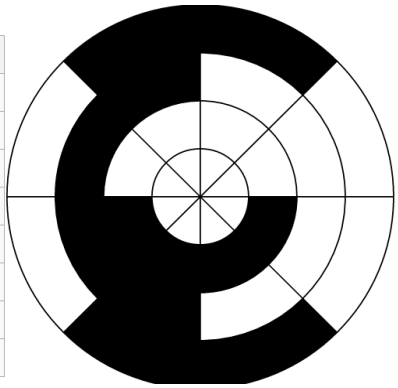


Figure 3.7: Gray binary system for 3-bit encoder.

3.1.3.3 Incremental rotary encoder

An incremental encoder provides a specified amount of pulses in one rotation of the encoder. The output can be a single line of pulses (an A channel) or two lines of pulses (an A and B channel) that are offset in order to determine rotation shown in Fig.3.8. This phasing between the two signals is called quadrature. The typical assembly of an incremental encoder consists of a spindle assembly, PCB, and cover. The PCB contains a sensor array that creates just two primary signals for the purpose of position and speed.

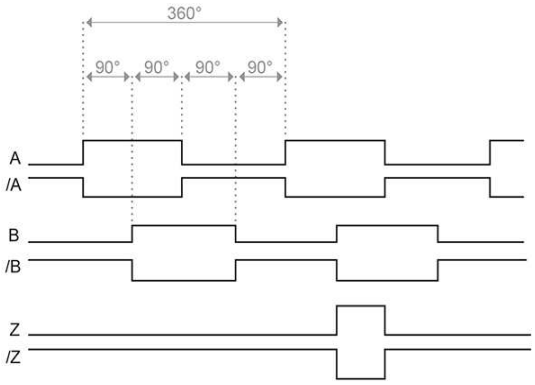


Figure 3.8: Two phase shifted signals of a incremental rotary encoder.

Optionally, additional signals can be provided: An index or Z channel can be

provided as one pulse per revolution signal for homing and pulse count verification on the A and or B channels. This index can be gated to either A or B in their various states. It can also be ungated and vary in width. Commutation (U, V, W) channels can also be provided on some encoders. These signals are aligned to the commutation windings found on servo motors. They also ensure that the drive or amplifier for those motors apply current to each winding in the correct sequence and at the correct level.

3.1.3.4 Sensorless

A sensorless control algorithm is much more complex than the more standard sensored control algorithm. It uses the back EMF of the unused phase of the motor in order to calculate the position of the rotor flux. The back EMF is induced in the unused coil of the motor as the rotor magnet passes by it. By reading the voltage caused by the back EMF a microcontroller can determine the position of the flux. Similar to the other control algorithm the microcontroller would use this position to determine which MOSFET switches to turn on and off.

One of the main flaws with this type of control algorithm is that the exact position of the rotor cannot be read at low or starting speed because there is not enough back EMF at this speed. This means that the performance of these motors is poor at lower speeds. This also makes starting the motor much more complex because the position at the beginning is not known which means additional software is needed to get the motor going. The benefit of this type of algorithm is the fact that it is more reliable due to the fact that it has no sensors and it is cheaper because it requires fewer components. Sensorless control also offers greater software customization which allows a single motor controller to be used on a wide variety of motors.

3.1.4 Mathematical model of the brushless motors

Brushless DC Motors are permanent magnet motors where the function of commutator and brushes were implemented by solid state switches. Brushless motors come in single-phase, 2-phase and 3-phase configurations. Corresponding to its type, the stator has the same number of windings. Out of these, 3-phase motors are the most

popular and widely used. Because of the special structure of the motor, it produces a trapezoidal or sinusoidal back emf and motor current generate a pulsating torque. Three phase BLDC motor equations, shown in Fig.3.9 are:

$$\begin{aligned} V_{an} &= i_a R_a + L_a \tfrac{di_a}{dt} - M_{ab} \tfrac{di_a}{dt} - M_{ac} \tfrac{di_a}{dt} + e_a \\ V_{bn} &= i_b R_b + L_b \tfrac{di_b}{dt} - M_{ba} \tfrac{di_b}{dt} - M_{bc} \tfrac{di_b}{dt} + e_b \\ V_{cn} &= i_c R_c + L_c \tfrac{di_c}{dt} - M_{ca} \tfrac{di_c}{dt} - M_{cb} \tfrac{di_c}{dt} + e_c \\ i_c &= -(i_a + i_b) \end{aligned} \tag{3.1}$$

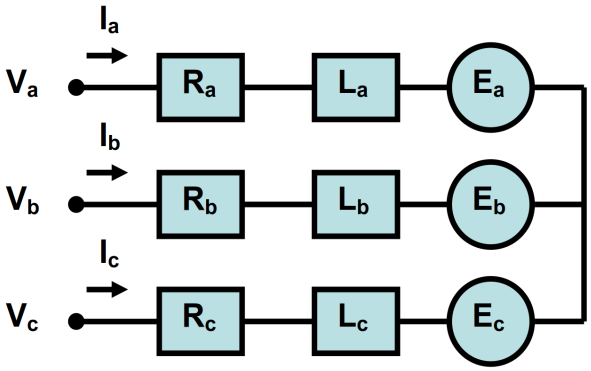


Figure 3.9: Brushless Motor phase circuit equivalent.

where:

- R: Stator resistance per phase, assumed to be equal for all phases.
- L: Stator inductance per phase, assumed to be equal for all phases.
- M: Mutual inductance between the phases.
- i_x: Stator current/phase.
- V_{xn}: are the respective phase x voltage of the winding with respect to neutral point voltage.

The stator self-inductances are independent of the rotor position, hence:
$L_x = L$.
And the mutual inductances will have the form of:
$M_{ab} = M_{ac} = M_{bc} = M_{ba} = M_{ca} = M_{cb} = \frac{M}{2}$.
Assuming three phase balanced system, all the phase resistances are equal:
$R_x = R$.

In most practical applications of BLDC motors, the stator windings are Y-connected in which there is no neutral point brought out so that the phase voltages are dif?cult to detect. Thus, the mathematical model based on phase voltage is not applicable in some cases. In contrast, the line voltage is easy to measure. It is approximately equal to the DC bus voltage when the relevant power transistors are turned on. Therefore, the mathematical model based on line voltage is more suited to the practical system. The line voltage equation can be obtained through subtraction calculation of the phase voltage equation as:

$$\begin{aligned}\begin{bmatrix} V_{ab} \\ V_{bc} \\ V_{ca} \end{bmatrix} &= \begin{bmatrix} R & -R & 0 \\ 0 & R & -R \\ -R & 0 & R \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + ... \\ ... &\begin{bmatrix} L-M & M-L & 0 \\ 0 & L-M & M-L \\ M-L & 0 & L-M \end{bmatrix} \frac{d}{dt} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} e_a - e_b \\ e_b - e_c \\ e_c - e_a \end{bmatrix}\end{aligned} \tag{3.2}$$

The back emf e_x of each phase can be written as:

$$\begin{aligned} e_a &= p\lambda\omega_m\phi(\theta_e) \\ e_b &= p\lambda\omega_m\phi(\theta_e + \tfrac{2\pi}{3}) \\ e_c &= p\lambda\omega_m\phi(\theta_e - \tfrac{2\pi}{3}) \end{aligned} \tag{3.3}$$

where:

- p: is the pole pairs in motor.
- λ: Amplitude of the flux induced by the permanent magnets of the rotor in the stator phases.

- ϕ: Normalized phase electromotive force.
- $\theta_e = p\theta_m$: electrical angel of rotor.
- ω_m: mechanical speed of rotor.
- $\omega_e = p\omega_m$: electrical speed of rotor.

The key to effective torque and speed control of a BLDC motor is based on relatively simple torque and back emf equations, which are similar to those of the DC motor. The generated electromagnetic torque is given by:

$$\mathrm{T}_e = \frac{[e_a i_a + e_b i_b + e_c i_c]}{\omega_m} (\text{in N.m}) \tag{3.4}$$

From mechanical point of view this produced electromagnetic torque is used to rotate the brushless motor. So we can write the mechanical formula of motion as:

$$\begin{aligned} &\tfrac{d}{dt}\omega_m = \tfrac{1}{J}(T_e - T_f - F\omega_m - T_m) \\ &\tfrac{d}{dt}\theta = \omega_m \end{aligned} \tag{3.5}$$

where:

- J: Combined inertia of rotor and load.
- F: Combined viscous friction of rotor and load.
- θ: Rotor angular position.
- T_m: Shaft mechanical load torque.
- T_f: Shaft static friction torque.
- ω_m: Angular velocity of the rotor (mechanical speed).

3.2 Topologies and Driving Algorithms

Brushless DC motors are becoming more common in a variety of motor applications such as fans, pumps, appliances, robotic automation, and automotive drives. The reasons for their increased popularity are better speed versus torque characteristics, high efficiency, long operating life, and noiseless operation. In addition to these advantages, the ratio of torque delivered to the size of the motor is higher, making it useful in applications where space and weight are critical factors. All this is possible by a very tight and efficient driver design. The basic structure of the brushless motor and the driver is shown in Fig.3.10. Almost all the old and new advanced brushless motor control algorithms use these kind of circuit topology. This topology uses three half-bridge structures for each phase of the brushless motor. By each half-bridge we can control the current flow in the each phase based on the other two phase's openness.

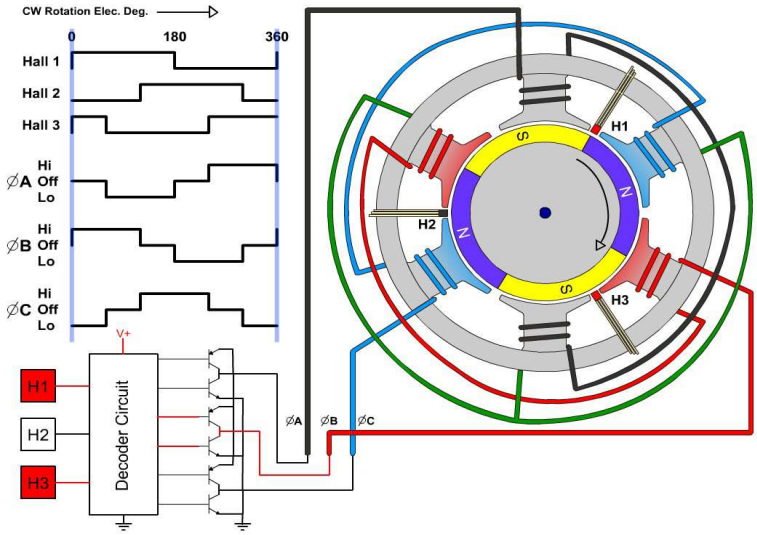


Figure 3.10: A 3-phase inside running BLDC with Hall sensors.

3.2.1 Controller

Brushless motors are needed in so many industrial equipments because of their high speed, high torque and good price. From a motor we need three kind of output, the

position, speed and torque. For these aims we need to control the brushless motor. Permanent magnet synchronous motors can be classified in many ways, but a couple are of interest, because they depend on back-EMF profiles: the brushless direct current (BLDC) motor and the permanent magnet synchronous motor (PMSM or BLAC). This terminology defines the shape of the back EMF of the synchronous motor. Both BLDC and PMSM motors have permanent magnets on the rotor, but differ in the flux distributions and back-EMF profiles, the differences are shown in table 3.1. To get the best performance out of the synchronous motor, it is important to identify the type of motor in order to apply the most appropriate type of control, as described in the next sections.

A motor controller is an electronic component of a drive system that converts a DC voltage into a three phase AC signal that can drive an AC motor. Based on the motor types and the efficiency required different control strategies have been developed. For complex and higher accuracy the Field Oriented Control (FOC) or Vector Control (VC) based on the differential equations of the system has been created. The FOC strategy has complex equations which for simplicity mostly it is used on PMSM (BLAC) motors.

Another type of control which is suitable for BLDC kinds is the 120°
trapezoidal or 6-step control. Since the current is bidirectional, each phase is broken into two steps per conducting interval. This is called six-step commutation. One commutation phase sequence option is AB-AC-BC-BA-CA-CB. Each conducting stage is called a step, and only two phases conduct current at any time, leaving the third phase floating. The unenergized winding can be used as a feedback control, which is the basis for sensorless control algorithms.

Most modern motor controllers are also equipped with regenerative capabilities. Regeneration allows the motor controller to gain back energy that it has already put out when the driver is braking. This is especially important in electric vehicles because of the limited amount of energy in the batteries.

3.2.2 6-step based algorithms

A BLDC motor revolves as a result of the interaction of its permanent magnet rotor with a magnetic field generated when a DC voltage is connected across a set of

BLDC	PMSM
Synchronous machine	Synchronous machine
Fed with direct currents	Fed with sinusoidal currents
Trapezoidal emf	Sinusoidal emf
Stator Flux position commutation each 60°	Continuous stator flux position variation
Only two phases ON at the same time	Possible to have three phases ON at the same time
Torque ripple at commutations	No torque ripple at commutations
Low order current harmonics in the audible range	Less harmonics due to sinusoidal excitation
Higher core losses due to harmonic content	Lower core loss
Less switching losses	Higher switching losses at the same switching freq
Control algorithms are relatively simple	Control algorithms are mathematically intensive

Table 3.1: Comparison of BLDC and PMSM Motors

stator coils. To maintain rotation, the orientation of the magnetic field in the stator has to be rotated sequentially. This is accomplished by connecting the DC voltage across the next set of stator coils as the rotor revolves. To maintain synchronization with the rotating stator magnetic field, the rotor position must be known at fixed angular intervals. BLDC motor are also known as electronically commutated (EC) motors and are driven by a DC-to- AC power inverter controlled with a dedicated MCU.

3.2.2.1 Driving and Excitation

The generation of the rotating magnetic field is implemented using a 6-transistor inverter bridge as shown in Fig.3.10. By controlling the commutation of six switching elements, the DC bus voltage can be applied across any combination of two stator coils of the BLDC motor. Fig.3.11 shows the 120-degree excitation method.

To drive the BLDC motor, the DC bus voltage is connected across two stator phases at any given time by turning on one upper and one lower transistor pair for 60 electrical degrees. For a six-element bridge, there are six distinct switching

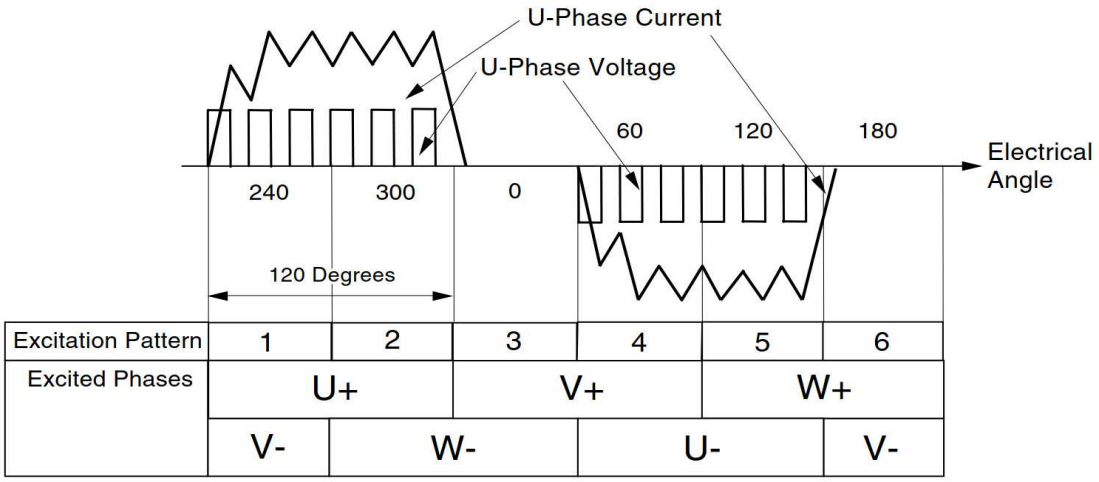


Figure 3.11: 120-Degree Excitation Pattern and U-Phase Current Waveform.

configurations: U+/V-, U+/W-, V+/W-, V+/U-, W+/U-, W+/V-. The BLDC motor rotates 360 electrical degrees if the six switching configurations are changed sequentially at 60-degree intervals in synchronization with the rotor position. After completing the six steps, the sequence is repeated. This method is called the 120-degree method because every switching element is turned on for 120 electrical degrees.

Fig.3.12 shows the six excitation patterns and the directions of the magnetic flux generated by the stator coils.

A BLDC motor revolves by the attraction and repulsion between the poles of the rotating magnetic field of the stator coils and the magnetic poles of the permanent magnet rotor. The torque developed is at maximum when the angle between the two magnetic poles is 90 degrees (the position of the permanent magnet poles changes as the rotor revolves).

In the 120-degree method, the commutation patterns are changed so that the angle between the magnetic pole of the rotor and rotating magnetic field of the stator is in a range of 60 to 120 degrees (the direction of the stator flux depends on how the coils are wound).

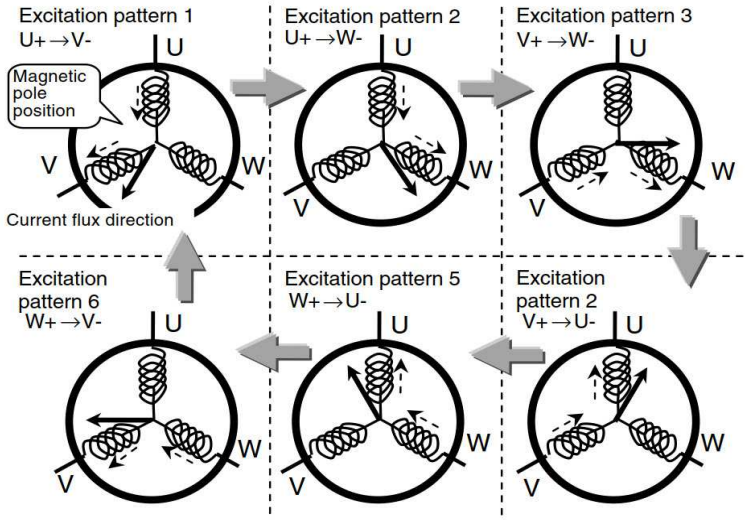


Figure 3.12: Excitation Patterns and Current Flux Directions.

3.2.2.2 Position Detection

For 120-degree or 6-step driving method the rotor position has to be known at 60-degree intervals. Fig.3.13 shows an example of a magnetic pole position detection using Hall effect sensors.

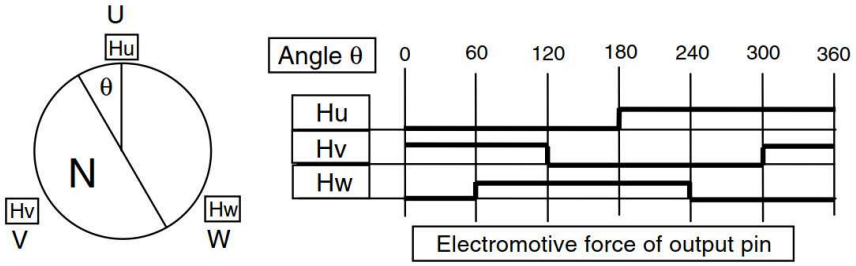


Figure 3.13: Hall IC.

To detect the rotor position, three Hall sensors are placed around the rotor perimeter at equal angular intervals. For a two-pole rotor, the Hall sensors are placed at 120 degrees. The logic state of each sensor changes every 180 degrees but

Excitation Pattern	1	2	3	4	5	6
Hu	H	L	L	L	H	H
Hv	H	H	H	L	L	L
Hw	L	L	H	H	H	L

Table 3.2: Relationship Between Hall ICs and Excitation Patterns

the combination pattern of the three sensors changes every 60-degree intervals. If the rotor has four poles, distinct positions can be detected at every 30 degrees physical angle: (physical angle = electrical angle/number of poles facing each other).

More commonly, a BLDC motor with four rotor poles has the Hall sensors spaced at 60-degree intervals so that the rotor position can be detected at 60 electrical degrees and 30 physical degrees. table 3.2 shows an example of an excitation pattern change based on the Hall sensor logic states.

3.2.2.3 Speed and Current Control

The revolution speed of the BLDC motor is directly proportional to the voltage applied to the stator coils. Turning on and off the switching elements with a pulse-width modulation (PWM) signal during the 120 degree excitation period can control the amplitude of the voltage. The chopping effect on the phase voltage controls the current through the coils and the torque developed in the rotor.To maintain a constant revolution speed, a proportional integral derivative PID) controller is use. The controller compares the speed measured by the Hall sensors with a "set" speed and adjusts the duty factor of the PWM signal accordingly by Space Vector Modulation. It is shown in Fig.3.14.b.

When the BLDC motor runs in the 120 degree conduction mode and the corresponding transient commutation process is ignored, the currents that have the same amplitude and the opposite direction only flow through two phase windings of Y connected motor at any time. Note that the functions of $\phi(\theta_e + \varphi)$ at the flat top position are apposite to each other. So equation 3.4 can be further simplified as:

$$T_e = 2p\lambda i_{phase} = K_t i_{phase} \tag{3.6}$$

where:

- K_t: the torque coefficient, that each company writes on datasheet of motor.
- i_{phase}: the steady phase current.

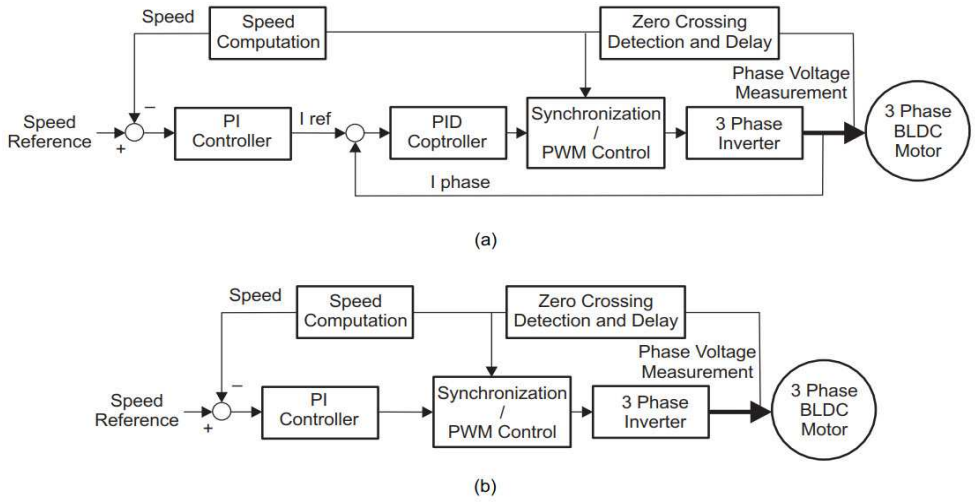


Figure 3.14: Speed and Current Control Loop Configurations for a BLDC Motor.

The BLDC motor is characterized by a two phase ON operation to control the inverter. In this control scheme, torque production follows the principle that current should flow in only two of the three phases at a time and that there should be no torque production in the region of the back EMF zero crossings. Fig.3.15 describes the electrical wave forms in the BLDC motor in the two phases ON operation. This control structure has several advantages:

- Only one current at a time needs to be controlled.
- Only one current sensor is necessary (or none for speed loop only, as detailed in the next sections).
- The positioning of the current sensor allows the use of low cost sensors as a shunt.

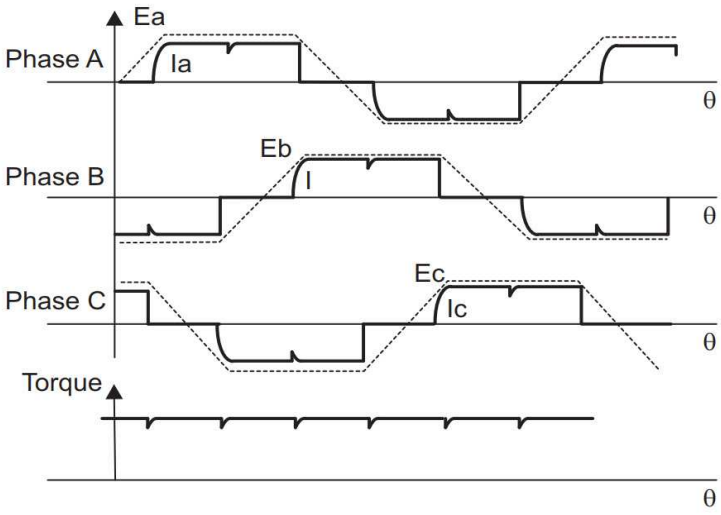


Figure 3.15: Electrical Waveforms in the Two Phase ON Operation and Torque Ripple.

The principle of the BLDC motor is, at all times, to energize the phase pair, which can produce the highest torque. To optimize this effect the back EMF shape is trapezoidal. The combination of a DC current with a trapezoidal back EMF makes it theoretically possible to produce a constant torque. In practice, the current cannot be established instantaneously in a motor phase; as a consequence the torque ripple is present at each 60 °phase commutation.

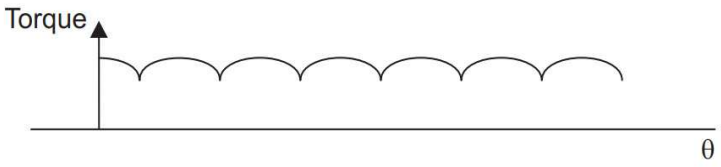


Figure 3.16: Torque Ripple in a Sinusoidal Motor Controlled as a BLDC.

If the motor used has a sinusoidal back EMF shape, this control can be applied but the produced torque shown in Fig.3.16, is :

- Not constant but made up from portions of a sine wave. This is due to its being

the combination of a trapezoidal current control strategy and of a sinusoidal back EMF. Bear in mind that a sinusoidal back EMF shape motor controlled with a sine wave strategy (three phase ON) produces a constant torque.

- The torque value produced is weaker.

3.2.2.4 System Topology

The BLDC motor control consists of generating DC currents in the motor phases. This control is subdivided into two independent operations: stator and rotor flux synchronization and control of the current value. Both operations are realized through the three phase inverter, which is shown in Fig.3.17.

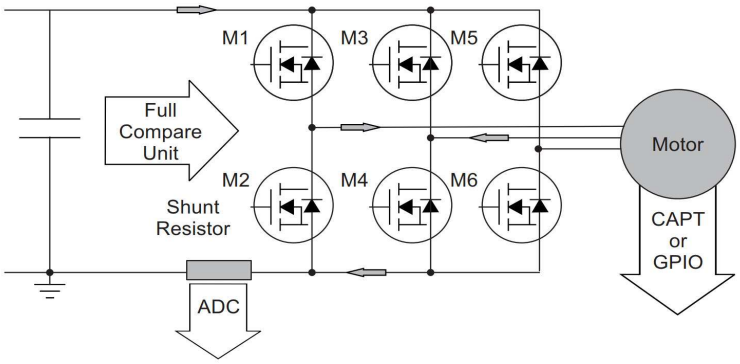


Figure 3.17: Three Phase Inverter.

The flux synchronization is derived from the position information coming from sensors, or from sensorless techniques. From the position, the controller determines the appropriate pair of transistors (M1 to M6) that must be driven. The regulation of the current to a fixed 60° reference can be realized in either of the two different modes:

- The Pulse Width Modulation (PWM) Mode:

  The supply voltage is chopped at a fixed frequency with a duty cycle depending on the current error. Therefore, both the current and the rate of change of current can be controlled. The two phase supply duration is limited by

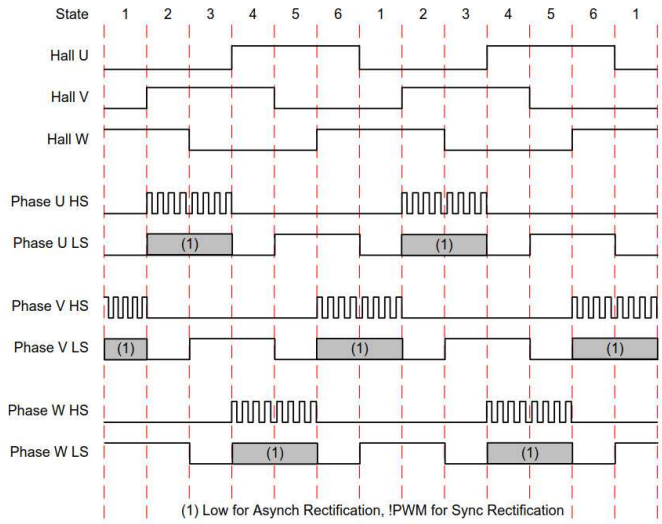


Figure 3.18: Standard 120° Commutation.

the two phase commutation angles. The main advantage of the PWM strategy is that the chopping frequency is a fixed parameter; hence, acoustic and electromagnetic noises are relatively easy to filter.

There are also two ways of handling the drive current switching: hard chopping and soft chopping. In the hard chopping technique, both phase transistors are driven by the same pulsed signal: the two transistors are switched-on and switched-off at the same time. The power electronics board is then easier to design and is also cheaper as it handles only three pulsed signals. A disadvantage of the hard chopping operation is that it increases the current ripple by a large factor in comparison with the soft chopping approach.

The soft chopping, which is shown in Fig.3.18, approach allows not only a control of the current and of the rate of change of the current but a minimization of the current ripple as well. In this soft chopping mode, the low side transistor is left ON during the phase supply and the high side transistor switches according to the pulsed signal. In this case, the power electronics board has to handle six PWM signals.

- The Hysteresis Mode:

In the hysteresis type current regulator, the power transistors are switched off and on according to whether the current is greater or less than a reference current. The error is used directly to control the states of the power transistors. The hysteresis controller is used to limit the phase current within a preset hysteresis band. As the supply voltage is fixed, the result is that the switching frequency varies as the current error varies. Therefore, the current chopping operation is not a fixed chopping frequency PWM technique. This method is more commonly implemented in drives where motor speed and load do not vary too much, so that the variation in switching frequency is small. Here again, both hard and soft chopping schemes are possible. Since the width of the tolerance band is a design parameter, this mode allows current control to be as precise as desired, but acoustic and electromagnetic noise are difficult to filter because of the varying switching frequency.

3.2.3 Sinusoidal Brushless motor control

Trapezoidal commutation is inadequate to provide smooth and precise motor control of brushless dc motors, particularly at low speeds. Sinusoidal commutation solves this problem. This is because the torque produced in a three phase brushless motor (with a sine wave back-EMF) is defined by the following equation:

$$\begin{cases} T_e = [i_a p\lambda \sin(\theta_e) + i_b p\lambda \sin(\theta_e + 120°) + i_c p\lambda \sin(\theta_e - 120°)] \\ i_a = I_0 \sin(\theta_e) \\ i_b = I_0 \sin(\theta_e + 120°) \\ i_c = -(i_a + i_b) = I_0 \sin(\theta_e - 120°) \\ \Rightarrow T_e = \frac{3}{2} p\lambda I_0 \end{cases} \tag{3.7}$$

Sinusoidally commutated brushless motor controllers attempt to drive the three motor windings with three currents that vary smoothly and sinusoidally as the motor turns. The relative phases of these currents are chosen so that they should result in a smoothly rotating current space vector that is always in the quadrature direction with respect to the rotor and has constant magnitude. This eliminates the torque ripple and commutation spikes associated with trapezoidal commutation.

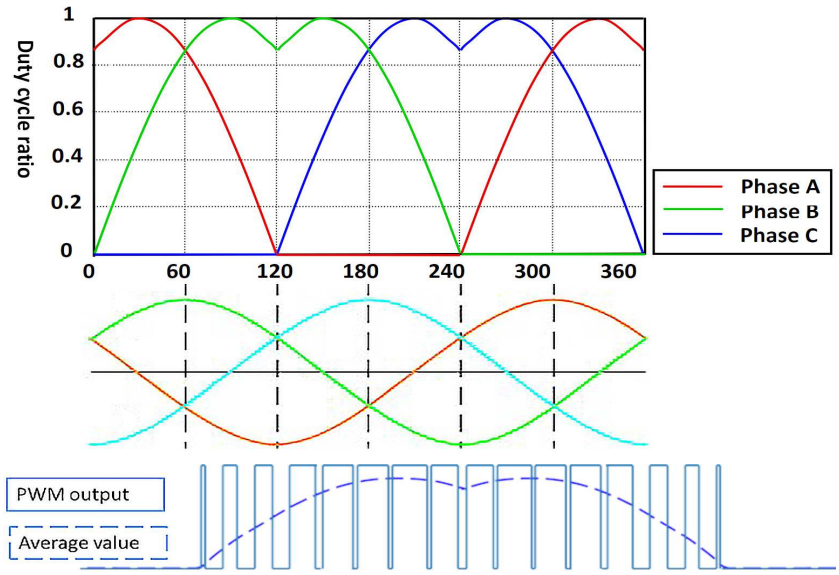


Figure 3.19: Three phase output and the sinusoidal current generated inside the phases.

In order to generate smooth sinusoidal modulation of the motor currents, which is shown in Fig.3.19 as the motor turns, an accurate measurement of rotor position is required. The Hall devices provide only a coarse measure of rotor position and are inadequate for this purpose. For this reason, angle feedback from an encoder, or similar device, is required.

Since the winding currents must combine to produce a smoothly rotating current space vector of constant magnitude, and because the stator windings are oriented 120 degrees apart from each other, currents in each winding must be sinusoidal and phase shifted by 120 degrees. Position information from the encoder is used to synthesize two sinusoids, one 120 degrees phase shifted from the other. These signals are then multiplied by the torque command so that the amplitudes of the sine waves are proportional to desired torque. The result is two sinusoidal current command signals appropriately phased to produce a rotating stator current space vector in the quadrature direction.

The sinusoidal current command signals are provided as inputs to a pair of P-I

controllers that regulate current in the two appropriate motor windings. The current in the third motor winding is the negative sum of the currents in the controlled windings and therefore cannot be separately controlled. The output from each P-I controller is fed to a PWM modulator and then to the output bridge and two motor terminals these voltages are shown in Fig.3.20. Voltage applied to the third motor terminal is derived as the negative sum of the signals applied to the first two windings, as appropriate for three sinusoidal voltages each separated by 120 degrees.

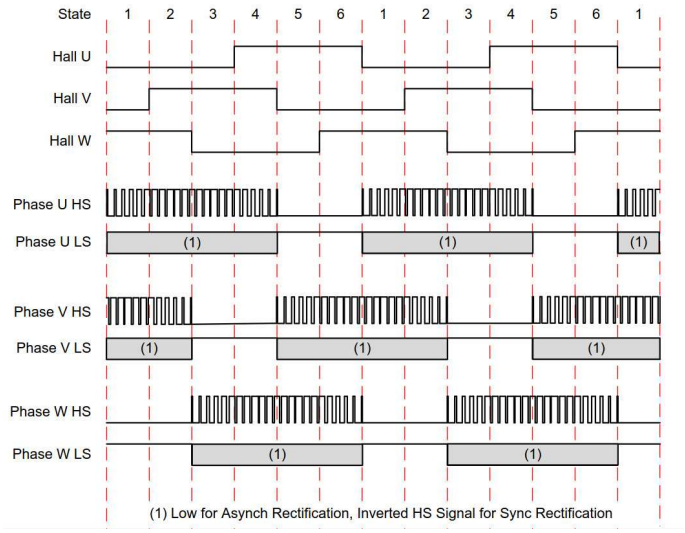


Figure 3.20: Sinusoidal commutation of PMSM with SPWM.

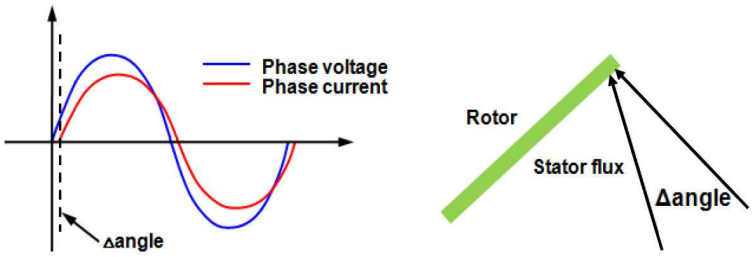


Figure 3.21: Advance angle between voltage and current.

To the extent that the actual output current waveform accurately tracks the sinusoidal current command signals, the resulting current space vector is smoothly rotating, constant in magnitude and oriented in the quadrate direction as desired.

Sinusoidal commutation results in smoothness of control that is generally unachievable with trapezoidal commutation. However, while it is very effective at low motor speeds, it tends to fall apart at high motor speeds. This is because as speed goes up the current loop controllers must track a sinusoidal signal of increasing frequency. At the same time they must overcome the motor back-EMF that also increases in amplitude and frequency as speed goes up.

Because the P-I controllers have limited gain and frequency response, the time-variant perturbations to the current control loop cause phase lag and gain error in the motor currents. Higher speeds result in larger errors. This perturbs the direction of the current space vector relative to the rotor, causing it to shift away from the quadrature direction. The phase voltage needs to be adjusted to keep the angle as close to 90° as possible between stator flux angle and rotor flux angle. This adjusted angle is called the advance angle , which is shown in Fig.3.21. When this happens, less torque is produced by a given amount of current and therefore more current is required to maintain torque. Efficiency deteriorates.

This degradation continues as speed increases. At some point motor current phase shift crosses through 90 degrees. When this happens torque is reduced to zero. With sinusoidal commutation, speeds above this point result in negative torque and are therefore not achievable.

3.2.4 Field Oriented Control

For Brushless DC motors, magnetic fields are generated by magnets mounted directly on the rotor and by coils in the stator. The stator windings generally come in a 3-phase configuration and are arranged to be 120 electrical degrees apart from each other. It is the sum of the force generated by these three phases that will ultimately generate useable motor rotation.

Depending on how the individual magnetic coils are phased, they can interact to create force that does not generate rotational torque, or they can create force which does drive rotation. These two different kinds of force are known as quadrature (Q)

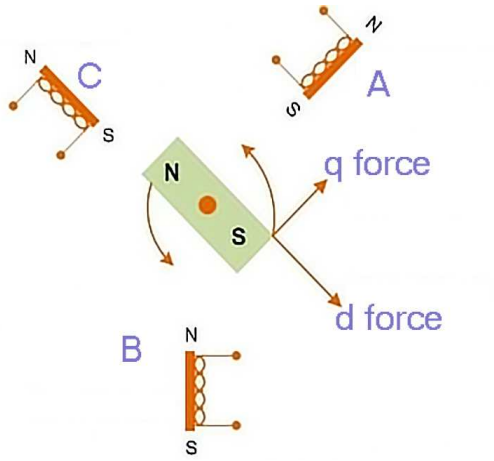


Figure 3.22: Two perpendicular magnetic force which generate power in motor.

and direct (D), with the useful quadrature forces (not to be confused with quadrature encoding scheme for position feedback devices) running perpendicular to the pole axis of the rotor, and the non-torque generating direct forces running parallel to the rotor's pole axis, which is shown in Fig.3.22. The trick to generating rotation is to maximize Q (quadrature) while minimizing D (direct) torque generation. If the rotor angle is measured using a Hall sensor or position encoder, the direction of the magnetic field from the rotor is known.

Six step commutation is a simple technique that reads Hall sensors and excites the coils in a specific sequence. The downside to this technique is that for many motors it gives up some efficiency and is not as smooth as more advanced techniques. This is because the output control signal for each coil changes abruptly when a new Hall state is read, which occurs every 60 electrical degrees, so the (Q) vector would be in 60 to 30 ° of the poles and this situation causes so much degradation in power efficiency and performance. That kind of performance is fine for simple spinning applications, or applications where the motor is geared way down. But for systems that need smoother motion and higher performance, two advanced techniques; sinusoidal control and Field oriented control, provide a jump in performance.

Field oriented control (FOC) is an important control approach for Brushless DC motors. It resembles sinusoidal commutation, but adds a major mathematical

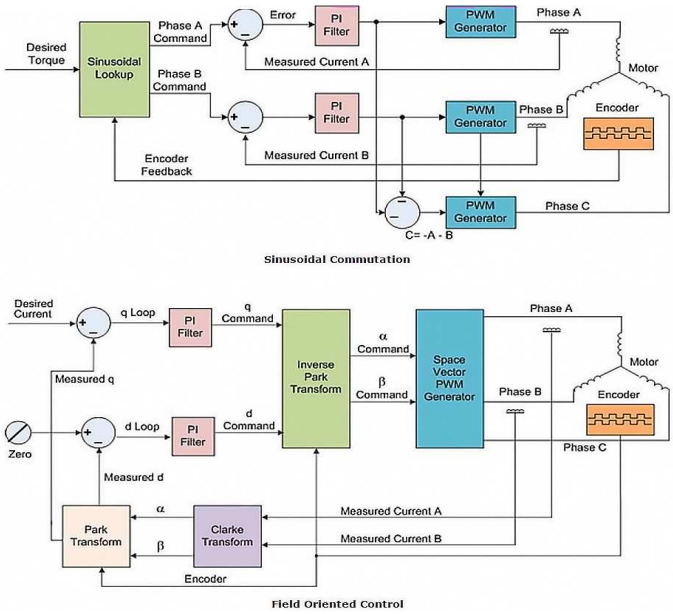


Figure 3.23: Sinusoidal and FOC block diagrams, which shows the differences of operations.

twist. The two controllers different schematic is shown in Fig.3.23. In the sinusoidal control approach, the torque command is 'vectorized' through a sinusoidal lookup table, thereby developing a separate command for each winding of the motor. As the rotor advances, the look-up angle advances in kind. Once the vectorized phase command is generated, it is passed on to a current loop, one for each winding, which attempts to keeps the actual winding current at the desired current value.

An important characteristic of this approach is that as the frequency of motor rotation increases, so does the challenge of maintaining the desired current. This is because the current loop is affected by the rotation frequency. Lag in the current loop, insignificant at low rotation speeds, generates increasing amounts of D (unwanted) torque at higher rotation speeds, resulting in a reduction of available torque.

The control scheme for field oriented control (FOC) differs in that the current loop occurs de-referenced from the motor's rotation. That is, independent of the motor's rotation. In the FOC approach there are two current loops, one for the

Q torque and another for the D torque. The Q torque loop is driven with the user's desired torque from the servo controller. The D loop is driven with an input command of zero, so as to minimize the unwanted direct torque component. The trick to making all of this work is math-intensive transform operations that convert the vectorized phase angle to, and from, the dereferenced D and Q reference frame. Known as Park and Clarke transforms, their practical implementation in Brushless DC and AC Induction drives is now commonplace due to the availability of low-cost, high-performance DSPs and microprocessors.

3.2.4.1 FOC theory

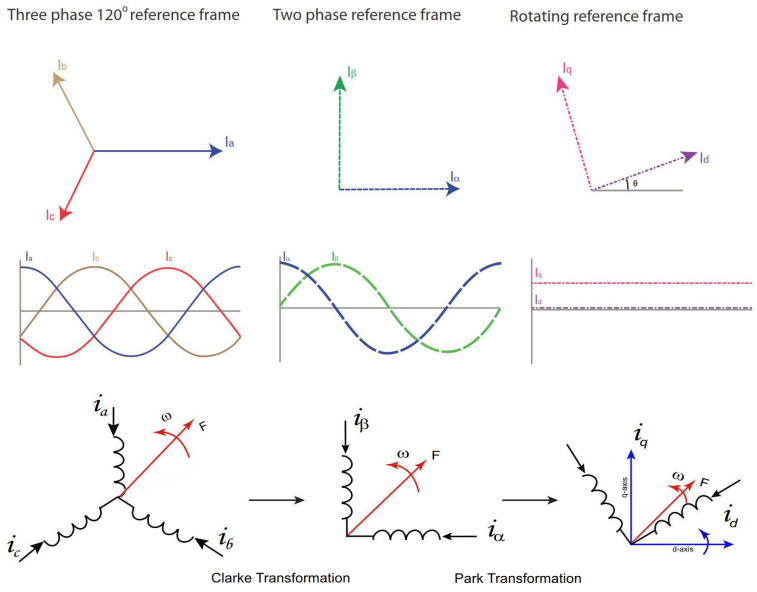


Figure 3.24: Transformations and Reference Frames and the Forward Transformations based on them.

In Field Oriented Control, motor currents and voltages are manipulated in the d-q reference frame of the rotor. This means that measured motor currents must be mathematically transformed from the three-phase static reference frame of the stator windings to the two axis rotating d-q reference frame, prior to processing by the PI controllers, these frames are shown in Fig.3.24. Similarly, the voltages to be

applied to the motor are transformed from the d-q frame of the rotor to the three phase reference frame of the stator before they can be used for PWM output.

The transformation from the 3-phase 120 degree reference frame to two axis orthogonal reference frame is known as Clarke transform. Similarly the transformation from two axis orthogonal reference frame to the two axis rotating reference frame is known as Park transform.

The measured motor currents are first translated from the 3-phase reference frame to the two axis orthogonal reference frame. The Clarke Transformation is expressed by the following equations:

$$\begin{cases} i_\alpha = i_a \\ i_\beta = \frac{(i_a+2i_b)}{\sqrt{3}} \\ where: i_a + i_b + i_c = 0 \end{cases} \tag{3.8}$$

The two axis orthogonal stationary reference frame quantities are then transformed by Park Transformation to rotating reference frame quantities. The transform is expressed by the following equations:

$$\begin{cases} i_d = i_\alpha \cos(\theta_e) + i_\beta \sin(\theta_e) \\ i_q = i_\beta \cos(\theta_e) - i_\alpha \sin(\theta_e) \end{cases} \tag{3.9}$$

The i_q produced would be subtracted from the desired i_q which is produced from desired T_e or desired ω_m from mechanical dynamics of the motor which shown in Fig.3.25 with details. The error will be fed into a PID controller to produce V_q and V_d. After all these voltages should be converted back to phase voltages which is done with Inverse Park-Clarke Transformation as:

$$\begin{cases} V_\alpha = V_d \cos(\theta_e) - V_q \sin(\theta_e) \\ V_\beta = V_q \cos(\theta_e) + V_d \sin(\theta_e) \end{cases}$$
$$\begin{cases} V_a = V_\alpha \\ V_b = \frac{-V_\alpha + \sqrt{3}V_\beta}{2} \\ V_c = \frac{-V_\alpha - \sqrt{3}V_\beta}{2} \end{cases} \tag{3.10}$$

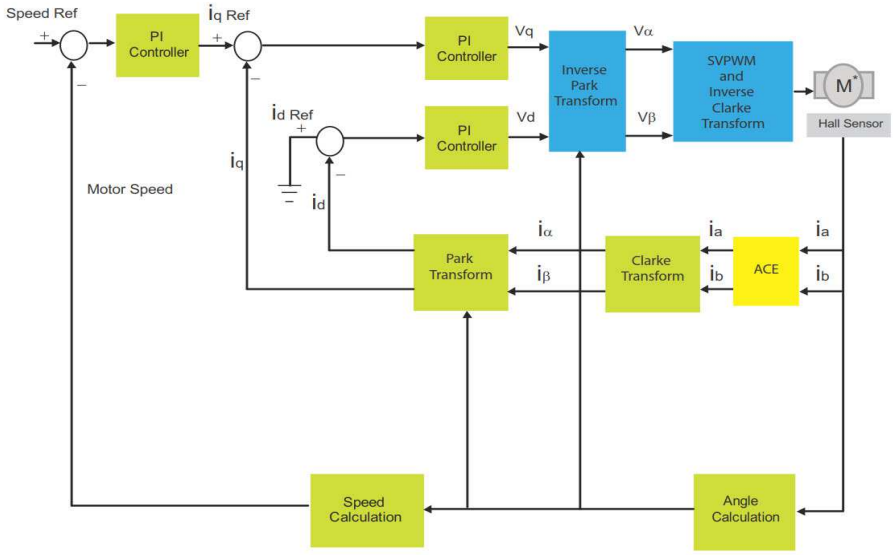


Figure 3.25: Detailed diagram of FOC.

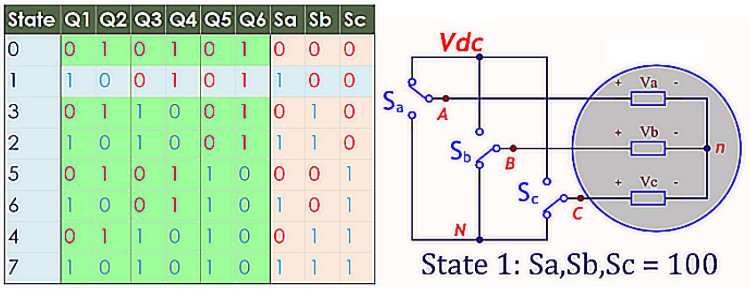
3.2.4.2 Space Vector Pulse Width Modulation

In SVPWM, the three phase variables are expressed in space vectors. Each desired voltage vector can be simulated by an averaging effect between two adjacent active vectors and a zero vector. Modulation strategy plays an important role in the minimization of harmonics and the switching losses in converters. The technique follows five set steps. First, phase angle and voltage reference vector is calculated based on input voltage components. Second, the modulation index is calculated and it is determined if the system is in the over modulation region. Third, the reference voltage vector section is found, as are the adjacent space vectors based on sector angle. Fourth, time intervals T_a, T_b, and T_o are calculated based on T_{source} and phase angle. Finally, for different switching states, the modulation times are calculated.

SVPWM is applied to output voltage and input current; the main objective being approximation of reference voltage vector utilizing eight switching patterns. The most common application of SVPWM is to create three-phase AC from a DC supply using multiple class-D amplifiers for use in driving motors. This technique generates less harmonic distortion in output voltage and input current to AC motor,

as well as provides advantages under unbalanced motor operation conditions leading it to be the PWM technique of choice.

One of the largest advantages of SVPWM algorithm for PWM is the increased efficiency of DC bus voltage use. SVPWM maximum peak fundamental magnitude of 90.6% of inverter capacity is 15.5% over SPWM.



State	Q1	Q2	Q3	Q4	Q5	Q6	Sa	Sb	Sc
0	0	1	0	1	0	1	0	0	0
1	1	0	0	1	0	1	1	0	0
3	0	1	1	0	0	1	0	1	0
2	1	0	1	0	0	1	1	1	0
5	0	1	0	1	1	0	0	0	1
6	1	0	0	1	1	0	1	0	1
4	0	1	1	0	1	0	0	1	1
7	1	0	1	0	1	0	1	1	1

Figure 3.26: Eight useful states of the inverter with a simplified circuit illustrating.

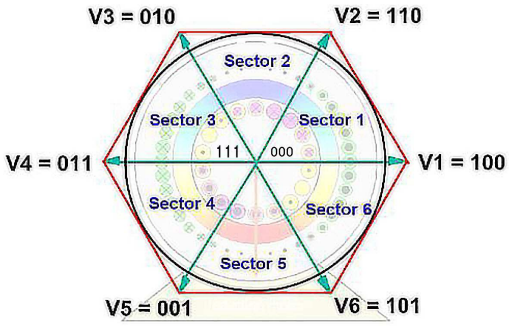


Figure 3.27: Six voltage vectors inside the generator resulting from state 1 to 6 of the inverter.

In SVPWM, the 3-phase inverter shown in Fig.3.26 is treated as a state machine. Two switches on the same leg cannot turn-on or turn- off at the same time, therefore, the state machine has eight states. Fig.3.26 shows these eight states with a simplified circuit illustrating state one. Six of the eight states result in voltage applied to the motor windings with two states resulting in zero voltage. Fig.3.27 illustrates directions and amplitudes of the space vector voltages applied to the motor

corresponding to the six active states. These vectors inform us of the direction that the inverter is attempting to establish a magnetic field inside the motor. To control the torque output of the motor, we need to create a smoothly rotating vector. The inverter must have the ability to establish a vector at any angle, not just the six vector shown in Fig.3.27.

Space vector representation of 3-phase quantities $x_a(t)$, $x_b(t)$, $x_c(t)$ is given by a vector in the α, β complex plane:

$$\overline{x} = x_\alpha + jx_\beta = x_a e^{j\omega t} + x_b e^{j(\omega t + \frac{2\pi}{3})} + x_c e^{j(\omega t - \frac{2\pi}{3})} \tag{3.11}$$

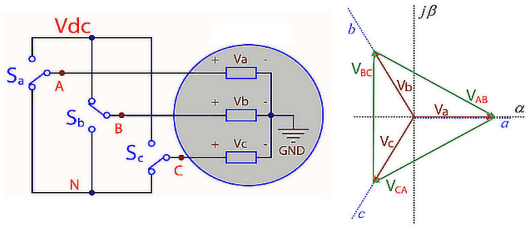


Figure 3.28: The 3-phase inverter and the α, β complex plane.

The α, β components of the space vector can also be achieved by Clarke transform:

$$\overline{x} = \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} = \frac{2}{3} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} x_a(t) \\ x_b(t) \\ x_c(t) \end{bmatrix} \tag{3.12}$$

The 3-phase inverter in Fig.3.28 is herein considered. As previously discussed, the inverter has eight useful switching states. Six of the eight states result in voltages applied to the motor windings with two states resulting in zero voltage.

The voltage output of the inverter can be represent by a space vector v in the α, β complex plane in Fig.3.28. This vector is calculated by phase voltages using eq.3.11:

$$\overline{v} = \frac{2}{3}(v_a + v_b\overline{a} + v_c\overline{a}^2);\ \overline{a} = e^{j(\frac{2\pi}{3})} \tag{3.13}$$

Where v_a, v_b, v_c are the output phase voltages of the inverter. According to KVL, we have:

$$\begin{cases} v_a = v_{AN} + v_N \\ v_b = v_{BN} + v_N \\ v_c = v_{CN} + v_N \end{cases} \tag{3.14}$$

Replacing these into eq.3.13:

$$\begin{aligned} \overline{v} &= \tfrac{2}{3}(v_{AN} + v_{BN}\overline{a} + v_{CN}\overline{a}^2 + v_N(1 + \overline{a} + \overline{a}^2)) \\ &= \tfrac{2}{3}(v_{AN} + v_{BN}\overline{a} + v_{CN}\overline{a}^2 + v_N(0)) \end{aligned} \tag{3.15}$$

With the state of each switch S_a; S_b; S_c equal to 0 or 1, we have: $v_{AN} = V_{dc}S_a$; $v_{BN} = V_{dc}S_b$; $v_{CN} = V_{dc}S_c$; and the eq.3.15 turns to:

$$\overline{v} = \frac{2}{3}V_{dc}(S_a + S_b\overline{a} + S_c\overline{a}^2) \tag{3.16}$$

State	S_a	S_b	S_c	Vector	$\frac{2}{3}V_{dc}(S_a + aS_b + a^2S_c)$	$\overline{v}$
0	0	0	0	$\overline{v}_0$	0	0
1	1	0	0	$\overline{v}_1$	$\frac{2}{3}V_{dc}$	$\frac{2}{3}V_{dc}e^{j0}$
2	1	1	0	$\overline{v}_2$	$\frac{2}{3}V_{dc}(1+a)$	$\frac{2}{3}V_{dc}e^{j(\pi/3)}$
3	0	1	0	$\overline{v}_3$	$\frac{2}{3}V_{dc}a$	$\frac{2}{3}V_{dc}e^{j(2\pi/3)}$
4	0	1	1	$\overline{v}_4$	$\frac{2}{3}V_{dc}(a+a^2)$	$\frac{2}{3}V_{dc}e^{j(3\pi/3)}$
5	0	0	1	$\overline{v}_5$	$\frac{2}{3}V_{dc}a^2$	$\frac{2}{3}V_{dc}e^{j(4\pi/3)}$
6	1	0	1	$\overline{v}_6$	$\frac{2}{3}V_{dc}(1+a^2)$	$\frac{2}{3}V_{dc}e^{j(5\pi/3)}$
7	1	1	1	$\overline{v}_7$	$\frac{2}{3}V_{dc}(1+a+a^2)$	0

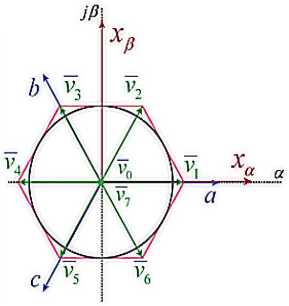


Figure 3.29: Voltage space vector corresponding to eight states of the inverter.

The output voltage space vector v corresponding to the eight states of the inverter is shown in Fig.3.29, can be generalized into:

$$\overline{v}_k = \begin{cases} \frac{2}{3}V_{dc}e^{j(k-1)\frac{\pi}{3}} & ; k = 1...6 \\ 0 & ; k = 0, 7 \end{cases} \tag{3.17}$$

In Vector control, the current controller calculates a reference voltage space vector $\overline{v}_{ref} = v_\alpha + jv_\beta$. Then, by SVPWM technique, the reference voltage vector v_{ref} is

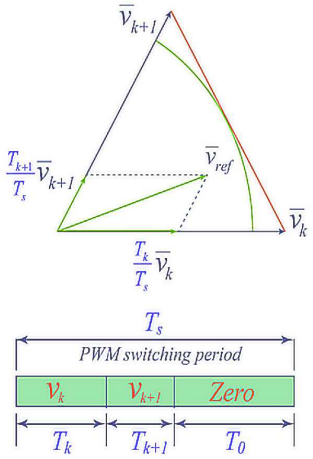


Figure 3.30: The reference voltage vector vref is resolved by time-averaging of the nearest two active switching vectors.

resolved by time averaging the nearest two active switching vectors which is shown in:

$$\overline{v}_{ref} = v_\alpha + jv_\beta = \frac{T_k}{T_s}\overline{v}_k + \frac{T_{k+1}}{T_s}\overline{v}_{k+1}; k = 1...6 \tag{3.18}$$

where:

- T_s: PWM switching period. $T_s = T_k + T_{k+1} + T_0$.
- T_k: Duration inverter output vector $\overline{v}_k$.
- T_{k+1}: Duration inverter output vector$\overline{v}_{k+1}$.
- T_0: Duration inverter output zero voltage.

Replacing v_k from eq.3.17 and solving for T_k and T_{k+1} we have:

$$\begin{bmatrix} T_k \\ T_{k+1} \end{bmatrix} = \frac{T_s\sqrt{3}}{V_{dc}} \begin{bmatrix} \sin(\frac{k\pi}{3}) & -\cos(\frac{k\pi}{3}) \\ -\sin(\frac{(k-1)\pi}{3}) & \cos(\frac{(k-1)\pi}{3}) \end{bmatrix} \begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix}; \; k = 1..6 \tag{3.19}$$

The order of applying switching space vectors is referred to as the modulation strategy. There are advanced modulation strategies for SVPWM, but the two most popular modulation strategies are: symmetrical PWM sequence and discontinuous space vector PWM sequence.

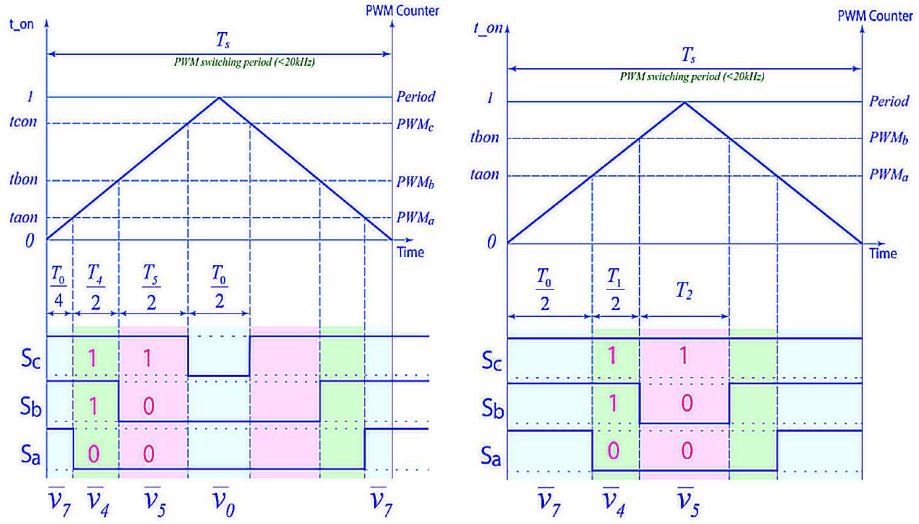


Figure 3.31: Symmetrical PWM sequence and Discontinue PWM sequence.

Symmetrical PWM Sequence (with v_{ref} in sector 4, $k = 4$) is depicted in Fig.3.31. The advantage of this modulation strategy is low THD resulting from its symmetry and because both zero vectors $\overline{v}_0$, $\overline{v}_7$ are used in one PWM cycle while there is only one leg of the inverter switching at a time.

Discontinuous Space Vector PWM sequence is depicted in Fig.3.31 (with v_{ref} in sector 1, $k = 1$). This modulation strategy keeps one of the three legs off during the entire 120 degree. The advantage of this modulation strategy is low switching loss due to fewer switching per PWM cycle. The drawback of this modulation strategy is high harmonic output.

The determination of location sector of v_{ref} can be done in many ways. We found the easiest way is to look at the sign of the voltage in phase a, b, c of the reference voltage vector v_{ref}. This method is found in Texas Instruments's reference code for C2000 MCU:
uint16 _t Sector = 0; Sector

	$v_\beta >= 0$			$v_\beta < 0$		
	$v_z < 0$	$v_z >= 0$		$v_z < 0$		$v_z >= 0$
		$v_\alpha >= 0$	$v_\alpha < 0$	$v_\alpha >= 0$	$v_\alpha < 0$	
Sector	1	2	3	6	5	4

Figure 3.32: Finding the sector based on v_α and v_β.

60 degree Sector determination
if (Va ¿ 0) Sector = 1;
if (Vb ¿ 0) Sector = Sector+2;
if (Vc ¿ 0) Sector = Sector+4;

Other method is to look at the sign of v_α; v_β and $v_z = v_\beta - \sqrt{3}v_\alpha$. This method is found is Freescale's reference code for Kinetis MCU shown in Fig.3.32.

3.2.5 ICs for BLDCs

Many instruments that are around us based on mechanical movements that came from a type of rotating electrical motors. Nowadays using brushless dc motors according to their high efficiency, high torque and speed is popular. Most of the market of BLDC motors are for lower power outputs around 30 to 70 watt. Many electronic companies such as ST and Texas Instrument produce ICs for motor control and these ICs are based on the demand of the market. Here we will table the ICs based on the topological placements as fully control, gate-drive and out-power stage MOSFETs or IGBTs.

3.3 Brushless Driver Implementation

The Driver developed is for the control of Trapezoidal Brushless Motor (BLDC) with Hall Sensors with six MOSFETs in "Three Half Bridges Configuration" with DC power supply up to 60 V and 3 A . The Driver has been designed to interface via I2C with an Higher System in which the speed control loop will be closed. The design of the system can be outlined in two parts:

Name	Brushless type	Topological Section	Voltage range	Current range	POWER
UCC3626 Texas Instruments 1999-2015	Both-T	1.Controller 6step	-	-	-
A4915 Allegro 2012-2015	Both-T	1.Controller 6step-Current 2.Gate-Drive	5-50	-	-
DRV8307 Texas Instruments 2014-2016	Both-T	1.Controller 6step	8.5-32	-	-
UC3625 Texas Instrument 1999-2013	Both-T	1.Controller 6step-Current 2.Gate-Drive	50	-	-
A4923 Allegro 2012-2016	Both-S	1.Controller Sinusoidal 2.Gate-Drive	36	-	-
FCM8201 Fairchild 2013-2016	Both-S	1.Controller Sinusoidal 2.Gate-Driver	30	-	-
L6235 ST 2003	Both	1.Controller 6step 2.Gate-Drive 3.Half-Bridges	8-52	2.8	35W
MC73110 PMDCorp 2007-2016	Both	1.Controller 6step-Sinusoidal Field Oriented	-	-	-
DRV8332 Texas Instrument 2010-2014	Both	1.Gate-Drive 2.Half-Bridges	50	15	5W
FAN7389 FairChild 2013	Both	1.Gate-Drive	600	650mA	
L6398 ST 2015	Both	1.Gate-Driver	600	250mA	800mW
STGIF5CH60TS-L	Both	1.Gates-Driver	600	8A	30W
STGIF10CH60TS-L	Both	1.Gates-Driver	600	15A	33W

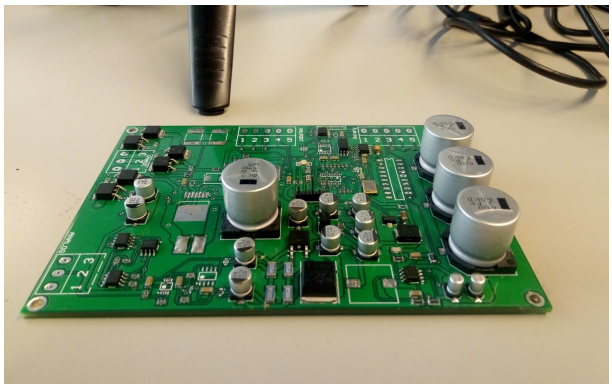


Figure 3.33: Developed Driver.

- The "power supply" section: where we will present the circuits developed in order to supply the "functional components" with the suitable power requirements.
- The "functional" section: where we will illustrate the functions that this board delivers together with a presentation of the electronic components that allow these functions to be accomplished.

For the parts interfacing via I2C with the higher system we will also presents the data exchange needed for the functioning. For the rest of the functional parts that doesn't directly interface with the Higher-System through I2C we will only present them shortly.

Chapter 4

Industrial Force and Torque Control

4.1 Introduction

Most robot controllers in use today are based on a position control scheme, which works well for free-motion tasks such as spray painting, but is inadequate for constrained motion tasks in which objects are grasped or contacted with tools. Even minute positional errors can cause excessive forces to develop between the manipulator and the contacted environment, leading to task failure or manipulator damage, especially in the presence of rigid environments. Positional errors are inevitable in practice and arise from a variety of sources including structural flexibility, finite actuator resolution, and geometric uncertainty in the environment. In order to be useful for constrained-motion tasks, robot manipulators must be designed with some form of compliance to compensate for misalignment.

A fundamental requirement for the success of a manipulation task is the capability to handle the physical contact between a robot and the environment. Force feedback and force control becomes mandatory to achieve a robust and versatile behavior of a robotic system in poorly structured environments as well as safe and dependable operation in the presence of humans. Robots using force, touch, distance, and visual feedback are expected to autonomously operate in unstructured environments other than the typical industrial shop floor.

Robot force control actually begins with remote manipulator and artificial arm control in the 1950s and1960s. The stability issues emerged immediately, but the strategy and task understanding issues remained submerged because people controlled these systems in "natural" ways. In the late 1960s and 1970s, the first computer controls of force feedback were attempted. Various approaches to creation of strategies emerged. All have depended on people to formulate the details and those which were tested experimentally encountered the stability problem. The first publications and experimental results concerning force/torque control can be dated down to the early 80s (and even earlier), the number of publications approaches its maximum from 2005 to 2009 and after that due to the recent improvements in microelectronics and computers power that makes it possible to implement the theoretical achievements in hardware.

4.2 Interaction Control

Control of the physical interaction between a robot manipulator and the environment is crucial for the successful execution of a number of practical tasks where the robot end-effector has to manipulate an object or perform some operation on a surface. Typical examples in industrial settings include polishing, deburring, machining or assembly. A complete classification of possible robot tasks, considering also nonindustrial applications, is practically infeasible in view of the large variety of cases that may occur, nor would such a classification be really useful to find a general strategy to control the interaction with the environment.

During the interaction, the environment sets constrains on geometric paths that can be followed by the end-effector. This situation is generally referred to as constrained motion. Successful execution of an interaction task with the environment by using motion control could be obtained only if the task were accurately planned and because detailed description of the environment is difficult to obtain so in such a case, the use of purely motion control turns out to be inadequate because the unavoidable modeling errors and uncertainties may cause a rise of the contact force, ultimately leading to an unstable behavior during the interaction. Interaction control strategies can be grouped in two categories: passive interaction control and

active interaction control.

4.3 Passive interaction control

In passive interaction control the trajectory of the robot end-effector is modified by the interaction forces due to the inherent compliance of the robot. The compliance may be due to the structural compliance of the links, joints, and end-effector, or to the compliance of the position servo. Soft robot arms with elastic joints or links are purposely designed for intrinsically safe interaction with humans. So in passive interaction, the end-effector of the robot is equipped with an auxiliary mechanical device, typically composed of springs and damper.

The passive approach to interaction control is very simple and cheap, because it does not require force/torque sensors; also, the preprogrammed trajectory of the end-effector must not be changed at execution time; moreover, the response of a passive compliance mechanism is much faster than active repositioning by a computer control algorithm. However, the use of passive compliance in industrial applications lacks flexibility, since for every robotic task a special-purpose compliant end-effector has to be designed and mounted. Also, it can only deal with small position and orientation deviations of the programmed trajectory. Finally, since no forces are measured, it cannot guarantee that high contact forces will never occur.

4.4 Active interaction control

In active interaction control, the compliance of the robotic system is mainly ensured by a purposely designed control system. This approach usually requires the measurement of the contact force and moment, which are fed back to the controller and used to modify or even generate online the desired trajectory of the robot end-effector.

Active interaction control may overcome the aforementioned disadvantages of passive interaction control, but it is usually slower, more expensive, and more sophisticated. To obtain a reasonable task execution speed and disturbance rejection capability, active interaction control has to be used in combination with some degree of passive compliance: feedback, by definition, always comes after a motion and

force error has occurred, hence some passive compliance is needed in order to keep the reaction forces below an acceptable threshold.

4.5 Force control

Active interaction control strategies can be grouped in two categories: those performing indirect force control and those performing direct force control. The main difference between two categories is that the former achieve force control via motion control both force and position information is required, without explicit closure of a force feedback loop, the latter, instead, offer the possibility of controlling the contact force to a desired value and only force information is required, thanks to the closure of a force feedback loop. To the first category belongs impedance control (or admittance control), where the deviation of the end-effector motion from the desired motion due to the interaction with the environment is related to the contact force through a mechanical impedance/admittance with adjustable parameters. A robot manipulator under impedance (or admittance) control is described by an equivalent mass-spring–damper system with adjustable parameters. This relationship is an impedance if the robot control reacts to the motion deviation by generating forces, while it corresponds to an admittance if the robot control reacts to interaction forces by imposing a deviation from the desired motion.

The measure difference between impedance control and force control is the commanded value: explicit force control requires commanded force, while impedance control require commanded position. In other words, impedance control achieve force control via motion control, without explicit closure of a force feedback loop; while the explicit force control instead, offer the possibility of controlling the contact force and moment to a desired value, thanks to the closure of a force feedback loop.

Special cases of impedance and admittance control are stiffness control and compliance control, respectively, where only the static relationship between the end-effector position and orientation deviation from the desired motion and the contact force and moment is considered. Therefore, robot force control involves integration of task goals like modeling the environment, position, velocity and force feedback,

and adjustment of the applied torque to the robot joints. Feedback of various measurement signals of the output of a robot (position, force, velocity) and the choice of command input signals to a robot result in different force control methods. These methods can be further categorized as indirect force control algorithms, direct force control strategies and advance force control.

4.6 Indirect Force Control

For a detailed analysis of interaction between the manipulator and environment it is worth considering the behavior of the system under a position control scheme when contact forces arise. Since these are naturally described in the operational space, it is convenient to refer to operational space control schemes.

Consider the manipulator dynamic model (2.31). Regarding the viscous friction torques as $\tau_f = F_v\dot{q}$, the model can be written as

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + F_v\dot{q} + G(q) = \tau - J^T(q)h_e \tag{4.1}$$

where h_e is the vector of contact forces exerted by the manipulator's end-effector on the environment.

4.6.1 Stiffness Control

Active stiffness control (see [65–80] and reference therein.) can be regarded as a programmable spring, since through a force feedback the stiffness of the closed-loop system is altered. In order to interaction control of manipulator with environment, the joint torques can be chosen as:

$$\tau = J^T(q)\alpha - K_D\dot{q} + G(q) \tag{4.2}$$

where the matrix J^T is needed to compute the resulting joint torques and the term g is needed to compensate for the static torques due to gravity; also the derivative term $K_D\dot{q}$, with k_D an $n \times n$ positive definite matrix gain, provides additional damping torques at the joints which are expected to improve the transient behavior of the system.

The control based on equation (4.2) is known as proportional-derivative (PD) control with gravity compensation.

In view of the partition of h_e in (2.30), it is appropriate to partition the vector α into its force and moment components

$$\alpha = \begin{bmatrix} \alpha_p \\ \alpha_o \end{bmatrix} \tag{4.3}$$

where α_p and α_o are 3×1 vectors which shall be chosen so as to provide a position and an orientation control action, respectively.

The end-effector position and orientation is described by a 6×1 vector $x_e = \begin{bmatrix} p_e^T & \varphi_e^T \end{bmatrix}^T$ where φ_e is a set of Euler angels extracted from R_e. Hence, a desired end-effector position and orientation can be assigned in terms of a constant vector x_d, corresponding to the position of the origin p_d and the rotation matrix R_d. The end-effector error can be denoted as $\Delta x_{de} = x_d - x_e$.

By choosing

$$\alpha = A^{-T}(\varphi_e) K_p \Delta x_{de} \tag{4.4}$$

where

$$A(\varphi_e) = \begin{pmatrix} I & 0 \\ 0 & T(\varphi_e) \end{pmatrix} \tag{4.5}$$

and

$$K_p = \begin{bmatrix} k_{p_p} & 0 \\ 0 & k_{p_o} \end{bmatrix} \tag{4.6}$$

where I is the 3×3 identity matrix, 0 is 3×3 null matrix, k_{p_p} and k_{p_o} is a suitable feedback matrix gain and T is the 3×3 matrix of mapping $w_e = T(\varphi_e)\dot{\varphi}_e$ depending on particular choice of the Euler angles.

In the absence of interaction with the environment (i. e., when $h_e = 0$), the stability of this system can be studied by introducing the following positive definite Lyapunov function candidate

$$v = \frac{1}{2}\dot{q}^T M(q)\dot{q} + \frac{1}{2}\Delta x_{de}^T K_p \Delta x_{de} \tag{4.7}$$

whose time derivative along the trajectories of the closed-loop system is the negative

semidefinite function

$$\dot{V} = -\dot{q}^T(F_\nu + k_D)\dot{q} \tag{4.8}$$

In the presence of a constant wrench h_e, using a similar Lyapunov argument, a different asymptotically stable equilibrium can be found, with a nonnull Δx_{de}. The new equilibrium is the solution of the equation

$$A^{-T}(\varphi_e)k_p\Delta x_{de} - h_e = 0 \tag{4.9}$$

which can be written in the form

$$\Delta x_{de} = k_p^{-1}A^T(\varphi_e)h_e \tag{4.10}$$

or equivalently

$$h_e = A^{-T}(\varphi_e)k_p\Delta x_{de} \tag{4.11}$$

Equation (4.11) shows that in the steady state the end-effector, under a proportional control action on the position and orientation error, behaves as a six-degree- of-freedom (DOF) spring in respect of the external force and moment h_e. Thus, the matrix k_p plays the role of an active stiffness, earning that it is possible to act on the elements of k_p so as to ensure a suitable elastic behavior of the end-effector during the interaction. Analogously, (4.10) represents a compliance relationship, where the matrix ${k_p}^{-1}$ plays the role of an active compliance. This approach, consisting of assigning a desired position and orientation and a suitable static relationship between the deviation of the end-effector position and orientation from the desired motion and the force exerted on the environment, is known as stiffness control.

The selection of the stiffness/compliance parameters is not easy, and strongly depends on the task to be executed. A higher value of the active stiffness means a higher accuracy of the position control at the expense of higher interaction forces. Hence, if it is expected to meet some physical constraint in a particular direction, the end-effector stiffness in that direction should be made low to ensure low interaction forces. Conversely, along the directions where physical constraints are not expected, the end-effector stiffness should be made high so as to follow closely the desired position. This allows discrepancies between the desired and achievable positions

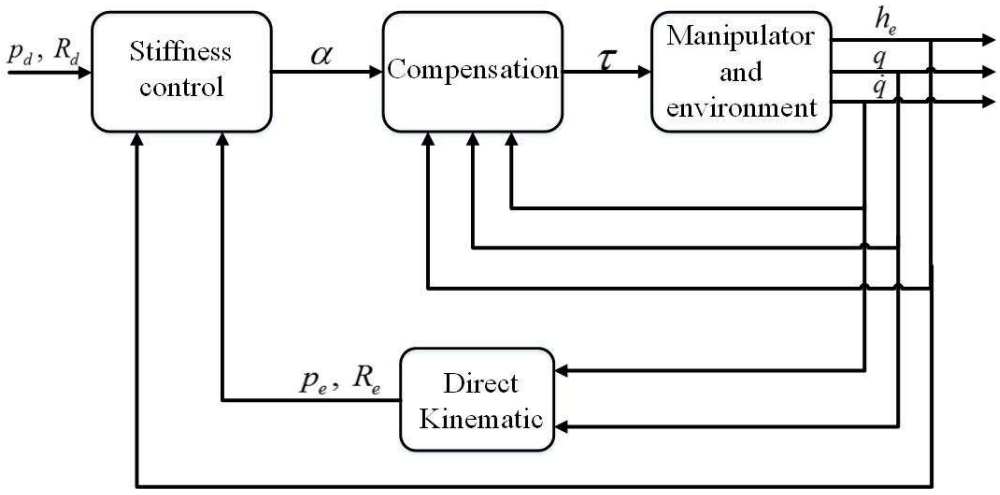


Figure 4.1: Active stiffness control.

due to the constraints imposed by the environment to be resolved without excessive contact forces and moments.

It must be pointed out, however, that a selective stiffness behavior along different directions cannot be effectively assigned in practice on the basis of (4.11). This can easily be understood by using the classical definition of a mechanical stiffness for two bodies connected by a 6-DOF spring, in terms of the linear mapping between the infinitesimal twist displacement of the two bodies at an unloaded equilibrium and the elastic wrench.

A block diagram of the resulting stiffness control with respect to controller 4.2 is sketched in Figure 4.1.

4.6.2 Impedance Control

Stiffness control is designed to achieve a desired static behavior of the interaction. In fact, the dynamics of the controlled system depends on that of the robot manipulator, which is nonlinear and coupled. A more demanding objective may be that of achieving a desired dynamic behavior for the end-effector, e.g., that of a second-order mechanical system with six degrees of freedom, characterized by a given mass, damping, and stiffness, known as mechanical impedance.

The fundamental philosophy of impedance control, according to Hogan ([81], [82] and [83]) is that the manipulator control system should be designed not to track a motion trajectory alone, but rather to regulate the mechanical impedance of the manipulator. Then, impedance control has been implemented in various forms, ([84–109]) depending on how the measured signals, i.e. velocity, position or force are used. In the following, we will design a basic impedance control.

A classical strategy to control a manipulator is inverse dynamics control, which is aimed at linearizing and decoupling the manipulator dynamics via feedback. Nonlinearities such as Coriolis and centrifugal torques, friction torques, and gravity torques can be merely canceled by adding these terms to the control input, while decoupling can be achieved by weighting the control input by the inertia matrix. According to this dynamic model-based compensation, the joint torques can be chosen as

$$\tau = M(q)\alpha + C(q,\dot{q})\dot{q} + F_\nu \dot{q} + G(q) + J^T(q)h_e \tag{4.12}$$

where α constitutes a new control input to be properly designed. Folding the control law (4.12) into the system model (4.1), and taking into account that $M(q)$ is always nonsingular, yields

$$\ddot{q} = \alpha \tag{4.13}$$

Since Equation (4.13) contains $\ddot{q}$, it is worth considering the time derivative of (2.14), i.e.

$$\dot{v}_e = J(q)\ddot{q} + \dot{J}(q,\dot{q})\dot{q} \tag{4.14}$$

which provides the relationship between the joint accelerations and the end-effector linear and angular accelerations.

At this point, the new control input η in (4.12) can be chosen as

$$\alpha = J^{-1}(q)(\eta - \dot{J}(q,\dot{q})\dot{q}) \tag{4.15}$$

where η attains the meaning of a resolved acceleration in terms of end-effector variables. In view of the partition of v_e in (4.14), it is appropriate to partition the vector

η into its linear and angular components

$$\eta = \begin{bmatrix} \eta_p \\ \eta_o \end{bmatrix}. \tag{4.16}$$

The end-effector position and orientation is described by a 6×1 vector $x_e = \begin{bmatrix} p_e^T & \varphi_e^T \end{bmatrix}^T$ where φ_e is a set of Euler angels extracted from R_e. Accordingly, the desired position and orientation is denoted as $x_d = \begin{bmatrix} p_d^T(t) & \varphi_d^T(t) \end{bmatrix}^T$. A position error between the desired and the actual end-effector position can be defined as

$$\Delta x_{de} = \begin{bmatrix} \Delta p_{de} \\ \Delta \varphi_{de} \end{bmatrix} = \begin{bmatrix} p_d - p_e \\ \varphi_d - \varphi_e \end{bmatrix} \tag{4.17}$$

Then, the resolved acceleration can be chosen as

$$\eta = A(\varphi_e)(\ddot{x}_d + {K_M}^{-1}(K_D \Delta \dot{x}_{de} + K_p \Delta x_{de} - A^T(\varphi_e) h_e)) + \dot{A}(\varphi_e, \dot{\varphi}_e)\dot{x}_e \tag{4.18}$$

with $A(\varphi_e)$ in (4.5), K_M, K_p and K_D define as

$$K_M = \begin{bmatrix} k_{M_p} & 0 \\ 0 & k_{M_o} \end{bmatrix}, \qquad K_D = \begin{bmatrix} k_{D_p} & 0 \\ 0 & k_{D_o} \end{bmatrix}, \qquad K_p = \begin{bmatrix} k_{p_p} & 0 \\ 0 & k_{p_o} \end{bmatrix} \tag{4.19}$$

where k_{M_p}, k_{M_o}, k_{D_p}, k_{D_o}, k_{p_p} and k_{p_0} are suitable feedback matrix gains. Substituting (4.18) into (4.15) gives the closed-loop dynamic behavior of the end-effector error

$$K_M \Delta \ddot{x}_{de} + K_D \Delta \dot{x}_{de} + K_p \Delta x_{de} = A^T(\varphi_e) h_e \tag{4.20}$$

In the case $h_e = 0$, the asymptotic stability of the equilibrium can be proven. The above equation establishes a relationship through a generalized active impedance between the contact force and moment h_e and the end-effector position and orientation error Δx_{de}. The equation (4.18), allow specifying the end-effector dynamics through a six-degree-of-freedom (six-DOF) impedance, where the transnational impedance is specified by k_{M_p}, k_{D_p} and k_{p_p}, and the rotational impedance is specified by k_{M_o}, k_{D_o} and k_{p_o}.

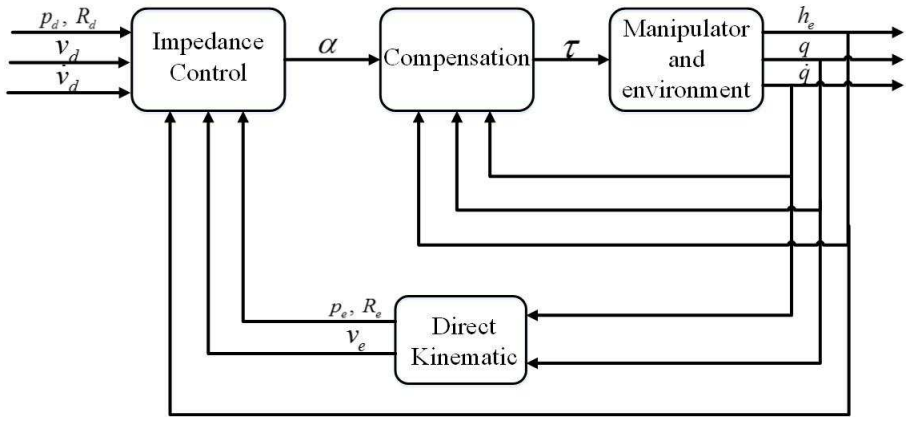


Figure 4.2: Impedance control.

This control scheme, in the absence of interaction, guarantees that the end-effector frame Σ_e asymptotically follows the desired frame Σ_d. In the presence of contact with the environment, a compliant dynamic behavior is imposed on the end-effector, and the contact wrench is bounded at the expense of a finite position and orientation displacement between σ_d and σ_e. Differently from stiffness control, a force/torque sensor is required for the measurement of the contact force and moment.

A block diagram of the resulting impedance control is sketched in Figure 4.2, With respect to the inverse dynamics scheme in (4.12), the position and orientation control is replaced with an impedance control which handles the measured contact force and moment and provides the resolved acceleration as in (4.15). Then, the inverse dynamics control law gives τ as in (4.12).

4.6.3 Admittance Control

The selection of good impedance parameters ensuring a satisfactory behavior is not an easy task. In fact, the dynamics of the closed-loop system is different in free space and during interaction. The control objectives are different as well, since motion tracking and disturbance rejection must be ensured in free space, while, during the interaction, the main goal is achieving a suitable compliant dynamic behavior for the end-effector. Notice also that the dynamics of the controlled system during the interaction depends on the dynamics of the environment.

In the previos section, equation (4.13) has been obtained under the assumption of perfect compensation of the terms in (4.1). This relies on the availability of an accurate dynamic model, as can be obtained via an identification procedure of the dynamic parameters. In case of imperfect compensation, a mismatching occurs which causes the presence of a disturbance term in (4.13), i.e.

$$\ddot{q} = \alpha - \delta \tag{4.21}$$

by stacking the equation (4.18) and accounting for the presence of a disturbance as in (4.21) yields

$$K_M \Delta\ddot{x}_{de} + K_D \Delta\dot{x}_{de} + K_p \Delta x_{de} = A^T(\varphi_e) h_e + K_M A^{-1}(\varphi_e) d \tag{4.22}$$

where

$$d = J(q)\delta \tag{4.23}$$

An effective disturbance rejection, at least at steady state, can be achieved by choosing a low weight for the matrix $K_p^{-1} K_M$ which corresponds to a stiff control action with an equivalent light mass at the end effector. Such a feature can eventually conflict with the desire of guaranteeing a compliant behavior when the end effector is in contact with a rather stiff environment.

A solution to this drawback can be devised by separating the motion control action from the impedance control action called admittance control ([110], [111], [112], [113], [114] and [115]) as follows. The motion control action is purposefully made stiff so as enhance disturbance rejection but, rather than ensuring tracking of the desired end-effector position and orientation, it shall ensure tracking of a reference position and orientation resulting from the impedance control action. In other words, the desired position and orientation together with the measured contact force and moment are input to the impedance equation which, via a suitable integration, generates the position and orientation to be used as a reference for the motion control action.

In order to realize the above solution, it is worth introducing a reference frame other than the desired frame specified by p_d and R_d. This frame is referred to as the compliant frame Σ_c, and is specified by a position vector p_c and a rotation

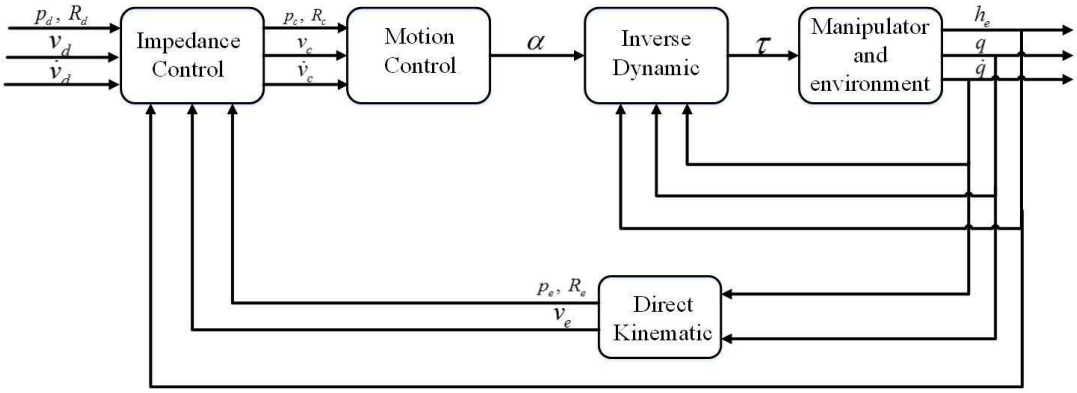


Figure 4.3: Impedance control with inner motion control loop (admittance control).

matrix R_c. In this way, the inverse dynamics motion control strategy can be still adopted as long as the actual end-effector position p_e and orientation R_e is taken to coincide with p_c and R_c in lieu of p_d and R_d, respectively. Accordingly, the actual end-effector linear velocity $\dot{p}_e$ and angular velocity w_e are taken to coincide with $\dot{p}_c$ and w_c, respectively.

A block diagram of the resulting scheme is sketched in Figure 4.3 and reveals the presence of an inner motion control loop with respect to the outer impedance control loop.

4.7 Direct Force Control

The above schemes implement an indirect force control, because the interaction force can be indirectly controlled by acting on the desired pose of the end-effector assigned to the motion control system. Interaction between manipulator and environment is anyhow directly influenced by compliance of the environment and by either compliance or impedance of the manipulator. Certain interaction tasks, however, do require the fulfillment of a precise value of the contact force. This would be possible, in theory, by tuning the active compliance control action and selecting a proper desired location for the end effector; such a strategy would be effective only on the assumption that an accurate estimate of the contact stiffness is available.

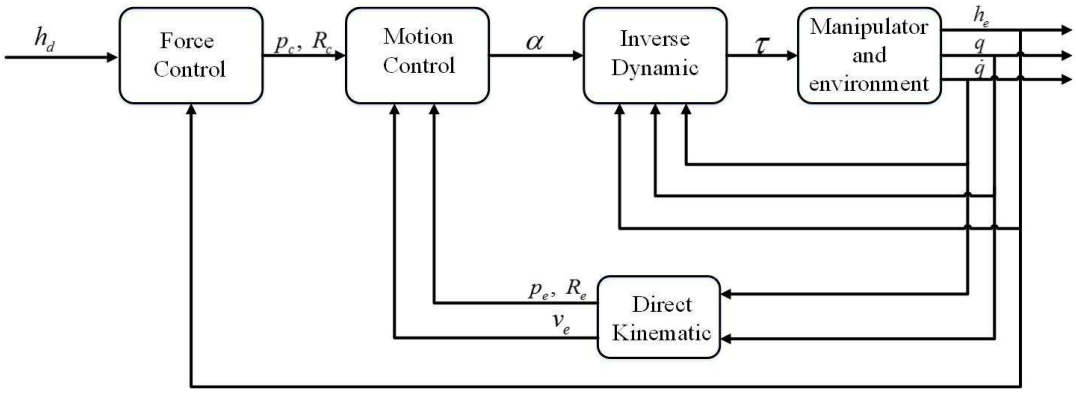


Figure 4.4: Explicit force control with inner motion control loop.

A radically different approach consists in designing a direct force control ([116–125]) which operates on a force error between the desired and the measured values. On the other hand, in the previous section it has been emphasized that even impedance control, which does not aim at achieving a desired force, needs contact force measurements to obtain a linear and decoupled end effector dynamics during the interaction. This is desirable in order to realize a compliant behavior only along those task directions that are actually constrained by the presence of the environment. Therefore, contact force measurements are fully exploited hereafter to design direct force control.

According to Volpe and Khosla ([117], [119]), who have investigated this subject and studied various methods, robot explicit force control includes two categories. One is force-based and the other is position-based explicit force control. In what follows only the force-based control strategy is going to be discussed.

4.7.1 Force Control

In order to design a force control, it is worth considering dynamic model-based compensation as in (4.12) with α in (4.15). The linear and angular accelerations can be respectively chosen as

$$\begin{aligned} \eta_p &= -K_{D_p}\dot{p}_e + K_{p_p}\Delta p_{ce} \\ \eta_o &= A(\varphi_e)(-K_{D_o}\dot{\varphi}_e + K_{p_o}\Delta\varphi_{ce}) + \dot{T}(\varphi_e,\dot{\varphi}_e)\dot{\varphi}_e \end{aligned} \tag{4.24}$$

where

$$\begin{aligned} p_c &= K_{p_p}^{-1}(K_{F_p}\Delta f + K_{I_p}\int_0^t \Delta f dc) \\ \varphi_c &= K_{p_o}^{-1}(K_{F_o}\Delta \mu + K_{I_o}\int_0^t \Delta \mu dc) \end{aligned} \tag{4.25}$$

where K_{F_p}, K_{F_o}, K_{I_p} and K_{I_o} are suitable positive definite matrix gains. In these equation $K_{p_p}^{-1}$ and K_{p0}^{-1} have been introduced in order to make the resulting force and moment control actions in (4.24 independent of the choice of the proportional gain on the position and orientation, respectively.

A block diagram of the resulting force and moment control scheme with dynamic model-based compensation is sketched in Figure 4.4 where the presence of the inner motion control loop has been evidenced.

4.8 Hybrid Position/Force Control

The hybrid control approach proposed by Raibert and Craig [126] in 1979 selects the directions in which the position of the end-effector of a manipulator should be controlled and the directions in which the force exerted by the end-effector on some object (or environment) should be controlled for a given task, and performs some control to make the position and force follow the given desired trajectories simultaneously. Usually, a control system based on this approach has two feedback loops for separated control of position and force. Current position in position-control directions and current force in force-controlled directions are measured. Corrections based on these measurements are applied by joint actuators to make the manipulator trace the desired position and force trajectories. Various generalization, extension, and modification of these this approach as well as its application to a number of robotic systems have been studied (see [67, 74, 101, 126–162] and reference therein).

It was shown, however, that this hybrid control may cause some unstable response [163]. One reason is that the dynamics of the manipulator is not considered rigorously. To overcome this difficulty and to formulate the force control problem more precisely, various researches have been done taking the arm dynamics into consideration: khatib et al. [128, 164] developed the operational space formulation, Yoshikawa et al. [165, 166] proposed the dynamic hybrid control approach, and McClamroch et al. [167, 168] and Kankaanranta et al. [169] proposed control algorithms

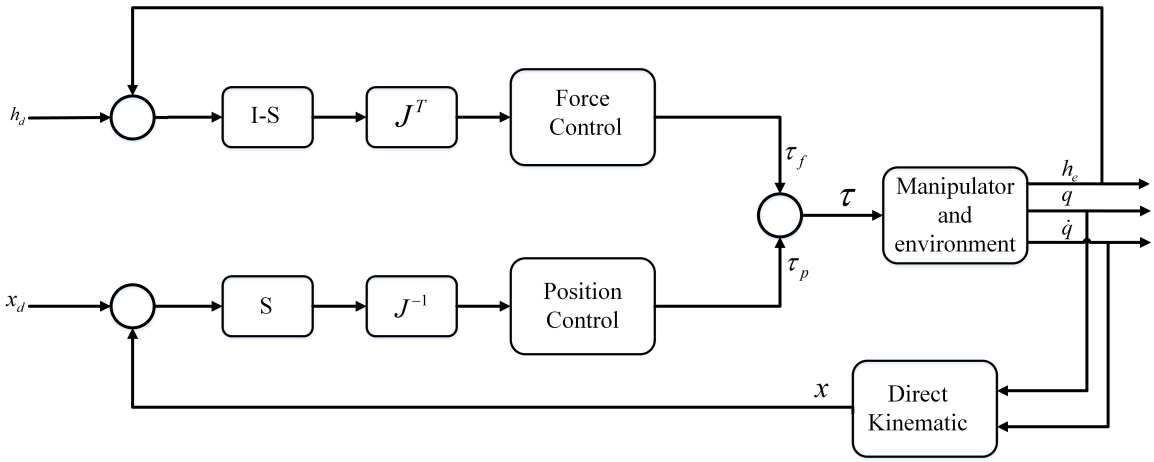


Figure 4.5: Hybrid force/position control.

similar to the dynamic hybrid control. Figure 4.5 shows the hybrid position/force control scheme. In this figure we see that the position control and force control can be separately considered. In Figure 4.5 ($S = diag(s_j)$, $j = 1, 2, ..., n$) is called the compliance selection matrix, n is the degree of freedom. The matrix S determines the subspace for which force or position are to be controlled, and s_j is selected as 1 or 0. When $s_j = 0$, the jth DOF must be force controlled, otherwise it must be position controlled. S matrix can be constant, can change with the configuration or can continuously change in time [170].

For each task configuration, a generalized surface can be defined in a constraint space with position constraints along the normals to this surface and force constraints along the tangents, which means, the end-effector can not move along the normals into the surface and can not exert forces along the tangents of the surface. These two types of constraints partition the degrees of freedom of possible end-effector motions into two orthogonal sets that must be either position or force controlled [127]. Utilizing this partition, S matrix is formed.

The command torque is

$$\tau = \tau_f + \tau_p \tag{4.26}$$

τ_f and τ_p are the command torques acting in position and force subspaces, respectively. In this way, position control and force control are decoupled. The control laws

for each can be designed independently, so that the different control performance requirements for the desired position and force trajectory tracking are simultaneously realized. Normally, the position control law in Figure 4.5 consists of a PD action, and the force control law consists of a PI action. This is because for the position control, a faster response is more desirable, and for the force control a smaller error is more preferable. Though, in some applications of hybrid position/force control, different force control algorithms are used for the force controlled subspace as explicit and implicit force control.

Chapter 5

Machine Vision in Robotics

After a brief silence in the 1990s, robotics has become reviving thanks to advancements in robotic autonomy made implementable through hyper fast, cheap, multi-threaded computing. Interestingly, different parts of the world have made leaps in different research directions, e.g., military robotics has made rapid progress in the US, while assistance robots have been dominant in Japan. However, the underlying technologies driving the various facets of robotics research are inter-transferable such that these advancements with seemingly different application areas are synergistically contributing to the rapid evolution of robotics as a whole.

This chapter introduces aspects of computer vision that are particularly relevant to visual servoing. Computer vision is the application of a computer system for receiving and processing visual information. It comprises two broad areas: image processing and image interpretation. The latter, often referred to as machine vision, is typically applied to part inspection and quality control, and involves the extraction of a small number of generally numeric features from the image. The former is the enhancement of images such that the resulting image more clearly depicts some desired characteristics for a human observer, and is commonly applied to medical and remote sensing imagery.

Recently, robotic vision as a research area has advanced dramatically, especially with the development of new range sensors. A multitude of techniques have been developed with impact on areas such as robotic navigation, scene/environment understanding, and visual learning, which are essential to the goal of bringing robots

into our daily lives [171].

Several industrial processes automated by robotic technology, require contact or interaction between the robot manipulator and its environment. The control of this interaction is crucial for the successful execution of many practical tasks where the robot's end-effector has to manipulate an object or perform some operation on a surface [172,173]. Typical industrial examples include drilling, polishing, deburring, machining or assembly. These manufacturing tasks require a given trajectory to be precisely followed over a large surface of unknown geometry. As an example, consider the cutting of arbitrary shapes over commercial plates and the cutting and welding operation over the surface of large containers, required for placing attachments like flanges and man-holes. In the case of robotic laser and plasma-cutting, the precision required in the placement of the cutting-tool over the surface is about $1[mm]$ for positioning and 1° for the orientation of the cutting-nozzle, with respect to the surface normal [174]. In these cases, the large variation in geometry of a commercial plate, caused by bending, or the relatively large tolerance in the fabrication of a large vessel, larger than the tolerance allowed for the placement of the cutting or welding tool, involves an intensive programming process, different for each piece.

When a robotic, vision-based controlled task is performed over large surfaces, the accuracy of the maneuver is limited by several factors. Among them, the resolution of the camera per unit physical space and the distortion of a previously planned path, when mapped over the arbitrary, curved surface. In the case of the tracking of a closed path, such a distortion may prevent the robot from achieving closure, which may represent an important drawback in tasks such as cutting. In order to solve the problem associated to the resolution of the camera per workspace size, several approaches can be followed. For instance, in [175] the use of a limited number of sensors combined with mirrors, is presented. Other approaches [176] consider the use of multiple cameras or the use of sensors mounted on top of pan/tilt units [177].

A particular goal of this chapter is to systematically examine the process of image formation and acquisition prior to it being digitally processed. Much computer vision literature places great emphasis on the digital processing stage while almost completely overlooking the issues involved in image formation and capture. Before an image becomes available in a frame store for computer analysis a number of complex, but frequently overlooked, transformations occur as depicted in Figure.5.2.

Figure 5.1: Some camera types used in image and video processing.

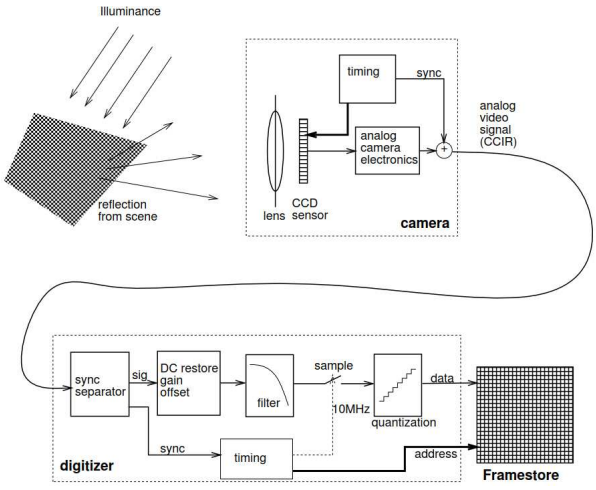


Figure 5.2: Steps involved in image processing.

These are due to illumination, lens, sensor, camera electronics and digitizer, and each can introduce artifacts into the final digital image. Particularly important for visual

servoing are the temporal characteristics of the camera, since the camera's shutter acts as the sampler in a visual control loop.

5.1 Pinhole camera model

In this section, we describe the image acquisition process known as the pinhole camera model, which is regularly employed as a basis in this thesis. More specifically, we first discuss the model that integrates the internal or intrinsic camera parameters, such as the focal length and the lens distortion. Secondly, we extend the presented simple camera model to integrate external or extrinsic camera parameters corresponding to the position and orientation of the camera.

5.1.1 Intrinsic camera parameters

The pinhole camera model defines the geometric relationship between a 3D point and its 2D corresponding projection onto the image plane. When using a pinhole camera model, this geometric mapping from 3D to 2D is called a perspective projection. We denote the center of the perspective projection (the point in which all the rays intersect) as the optical center or camera center and the line perpendicular to the image plane passing through the optical center as the optical axis (see Figure.5.3). Additionally, the intersection point of the image plane with the optical axis is called the principal point. The pinhole camera that models a perspective projection of 3D points onto the image plane can be described as follows.

A. Perspective projection using homogeneous coordinates

The projection of a 3D world point $(X, Y, Z)^T$ onto the image plane at pixel position $(u, v)^T$ can be written as:

$$\begin{cases} u = \frac{Xf}{Z} \\ v = \frac{Yf}{Z} \end{cases} \tag{5.1}$$

where f denotes the focal length. To avoid such a non-linear division operation, the previous relation can be reformulated using the projective geometry framework, as:

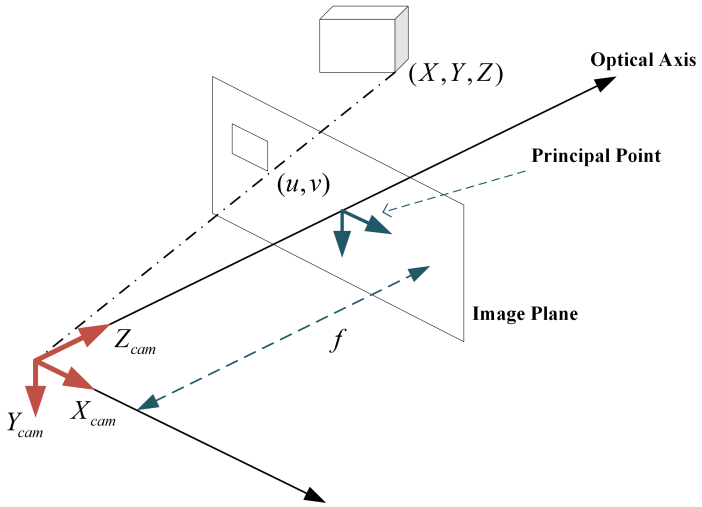


Figure 5.3: The ideal pinhole camera model describes the relationship between a 3D point $(X, Y, Z)^T$ and its corresponding 2D projection $(u, v)^T$ onto the image plane.

$$(\lambda u, \lambda v, \lambda)^T = (Xf, Yf, Z)^T \tag{5.2}$$

This relation can be the expressed in matrix notation by:

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{5.3}$$

where $\lambda = Z$ is the homogeneous scaling factor.

B. Principal-point offset

Most of the current imaging systems define the origin of the pixel coordinate system at the top-left pixel of the image. However, it was previously assumed that the origin of the pixel coordinate system corresponds to the principal point $(o_x, o_y)^T$, located at the center of the image (see Figure.5.4). A conversion of coordinate systems is thus necessary. Using homogeneous coordinates, the principal-point position can be readily integrated into the projection matrix. The perspective projection equation becomes now:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f & 0 & o_x & 0 \\ 0 & f & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{5.4}$$

where $(x, y)^T$ is the image coordinate system.

C. Image-sensor characteristics

To derive the relation described by Equation.5.4, it was implicitly assumed that the pixels of the image sensor are square, i.e., aspect ratio is $1:1$ and pixels are not skewed. However, both assumptions may not always be valid. First, for example, an NTSC TV system defines non-square pixels with an aspect ratio of $10:11$. In practice, the pixel aspect ratio is often provided by the image-sensor manufacturer. Second, pixels can potentially be skewed, especially in the case that the image is acquired by a frame grabber. In this particular case, the pixel grid may be skewed due to an inaccurate synchronization of the pixel-sampling process. Both previously mentioned imperfections of the imaging system can be taken into account in the camera model, using the parameters η and τ , which model the pixel aspect ratio and skew of the pixels, respectively (see Figure.5.4). The projection mapping can be now updated as:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f & \tau & o_x & 0 \\ 0 & \eta f & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} K & 0_3 \end{bmatrix} P \tag{5.5}$$

with $P = (X, Y, Z, 1)^T$ being a $3D$ point defined with homogeneous coordinates. In practice, when employing recent digital cameras, it can be safely assumed that pixels are square ($\eta = 1$) and non-skewed ($\tau = 0$). The projection matrix that incorporates the intrinsic parameters is denoted as K throughout this thesis. The all zero element vector is denoted by 0_3.

D. Radial lens distortion

Real camera lenses typically suffer from non-linear lens distortion. In practice, radial lens distortion causes straight lines to be mapped as curved lines. As seen in

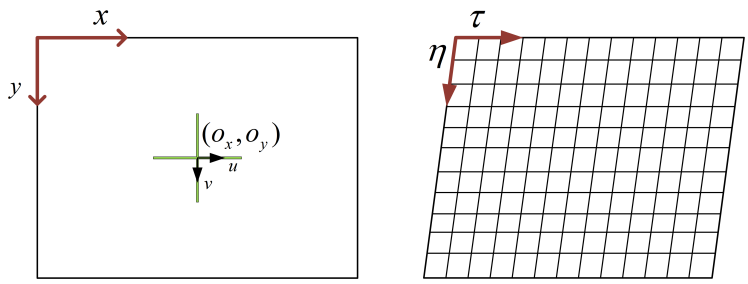


Figure 5.4: (left) The image (x, y) and camera (u, v) coordinate system. (right) Non-ideal image sensor with non-square, skewed pixels.

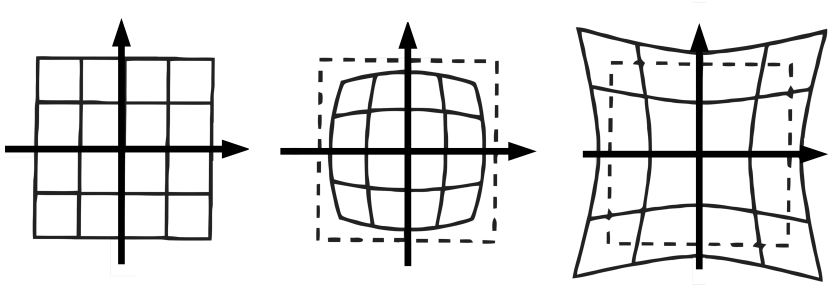


Figure 5.5: Real camera lenses suffer from radial lens distortion that causes straight lines to be bended. Pixel grid of an (left) undistorted and two (right) distorted images.

Figure.5.5, the radial lens distortion appears more visible at the image edges, where the radial distance is high. A standard technique to model the radial lens can be described as follows.

Let $(x_u, y_u)^T$ and $(x_d, y_d)^T$ be the corrected and the measured distorted pixel positions, respectively. The relation between an undistorted and distorted pixel can be modeled with a polynomial function and can be written as:

$$\begin{pmatrix} x_u - o_x \\ y_u - o_y \end{pmatrix} = L(r_d) \begin{pmatrix} x_d - o_x \\ y_d - o_y \end{pmatrix} \tag{5.6}$$

where:

$$L(r_d) = 1 + k_1 r_d^2; r_d^2 = (x_d - o_x)^2 + (y_d - o_y)^2 \tag{5.7}$$

In the case $k_1 = 0$, it can be noted that $x_u = x_d$ and $y_u = y_d$, which corresponds to the absence of radial lens distortion.

It should be noted that Equation.5.6 provides the correct pixel position using a function of the distorted pixel position. However, to generate an undistorted image, it would be more convenient to base the function $L(r)$ on the undistorted pixel position. This technique is usually known as the inverse mapping method. The inverse mapping technique consists of scanning each pixel in the output image and re-sampling and interpolating the correct pixel from the input image. To perform an inverse mapping, the inversion of the radial lens distortion model is necessary and can be described as follows. First, similar to the second part of Equation.5.7, we define:

$$r_u^2 = (x_u - o_x)^2 + (y_u - o_y)^2 \tag{5.8}$$

Then, taking the norm of Equation.5.6, it can be derived that:

$$(x_u - o_x)^2 + (y_u - o_y)^2 = L(r_d).((x_d - o_x)^2 + (y_d - o_y)^2) \tag{5.9}$$

which is equivalent to:

$$r_u = L(r_d).r_d \tag{5.10}$$

When taking into account Equation.5.7, this equation can be rewritten as a cubic polynomial:

$$r_d^3 + \frac{1}{k_1} r_d - \frac{r_u}{k_1} = 0 \tag{5.11}$$

The inverted lens distortion function can be derived by substituting Equation.5.10 into Equation.5.6 and developing it from the right-hand side:

$$\begin{pmatrix} x_d - o_x \\ y_d - o_y \end{pmatrix} = \frac{r_d}{r_u} \begin{pmatrix} x_u - o_x \\ y_u - o_y \end{pmatrix} \tag{5.12}$$

where r_d can be calculated by solving the cubic polynomial function of Equation.5.11. This polynomial can be solved using Cardano's method, by first calculating the dis-

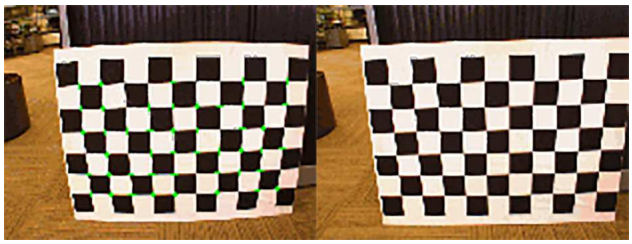


Figure 5.6: (left) Distorted image. (right) Corresponding corrected image using the inverted mapping method.

criminant Δ defined as $\Delta = q^2 + (4/27)p^3$ where $p = 1/k1$ and $q = -ru/k1$. Depending on the sign of the discriminant, three sets of solutions are possible.

If $\Delta > 0$, then the equation has one real root rd1 defined as:

$$r_{d1} = \sqrt[3]{\frac{-q + \sqrt{\Delta}}{2}} + \sqrt[3]{\frac{-q - \sqrt{\Delta}}{2}} \tag{5.13}$$

If $\Delta < 0$, then the equation has three real roots r_{dk} defined by:

$$r_{dk} = 2\sqrt{\frac{-p}{3}} \cos\left(\frac{\arccos\left(\frac{-q}{2}\sqrt{\frac{27}{-p^3}}\right) + 2k\pi}{3}\right) \tag{5.14}$$

for $k = 0, 1, 2$; where an appropriate solution r_{dk} should be selected such that $r_{dk} > 0$ and $r_{dk} < r_{uk}$. However, only one single radius corresponds to the practical solution. Therefore, the second case $\Delta < 0$ should not be encountered. The third case with $\Delta = 0$ is also impractical. In practice, we have noticed that, indeed, these second and third cases never occur.

As an example, Figure.5.6 depicts a distorted image and the corresponding corrected image using the inverted mapping method, with $\Delta > 0$.

Estimation of the distortion parameters The discussed lens-distortion correction method requires knowledge of the lens parameters, i.e., k_1 and $(o_x, o_y)^T$. The estimation of the distortion parameters can be performed by minimizing a cost function that measures the curvature of lines in the distorted image. To measure this curvature, a practical solution is to detect feature points belonging to the same

line on a calibration rig, e.g., a checkerboard calibration pattern (see Figure.5.6). Each point belonging to the same line in the distorted image forms a bended line instead of a straight line [178]. By comparing the deviation of the bended line from the theoretical straight line model, the distortion parameters can be calculated.

5.1.2 Extrinsic parameters

As opposed to the intrinsic parameters that describe internal parameters of the camera (focal distance, radial lens parameters), the extrinsic parameters indicate the external position and orientation of the camera in the $3D$ world. Mathematically, the position and orientation of the camera is defined by a 3×1 vector C and by a 3×3 rotation matrix R (see Figure.5.7).

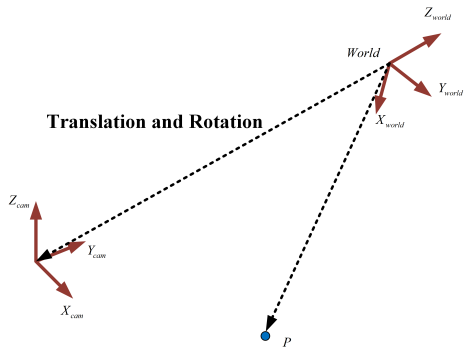


Figure 5.7: The relationship between the camera and world coordinate system is defined by the camera center C and the rotation R of the camera.

To obtain the pixel position $p = (x, y, 1)^T$ of a $3D$-world homogeneous point P, the camera should be first translated to the world coordinate origin and second, rotated. This can be mathematically written as:

$$\lambda p = [K|0_3] \begin{bmatrix} R & 0 \\ 0_3 & 1 \end{bmatrix} \begin{bmatrix} 0_3^T & -C \\ 0 & 1 \end{bmatrix} P \tag{5.15}$$

Alternatively, when combining matrices, Equation.5.15 can be reformulated as:

$$\lambda p = [K|0_3] \begin{bmatrix} R & -RC \\ 0_3^T & 1 \end{bmatrix} P = KR \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} - KRC \tag{5.16}$$

Back-projection of a 2D point to 3D

Previously, the process of projecting a $3D$ point onto the $2D$ image plane was described. We now present how a $2D$ point can be back-projected to the $3D$ space and derive the corresponding coordinates. Considering a $2D$ point p in an image, there exists a collection of $3D$ points that are mapped and projected onto the same point p. This collection of $3D$ points constitutes a ray connecting the camera center $C = (C_x, C_y, C_z)^T$ and $p = (x, y, 1)^T$. From Equation.5.16, the ray $P(\lambda)$ associated to a pixel $p = (x, y, 1)^T$ can be defined as:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \underbrace{C + \lambda R^{-1} K^{-1} p}_{ray, P(\lambda)} \tag{5.17}$$

where λ is the positive scaling factor defining the position of the $3D$ point on the ray. In the case Z is known, it is possible to obtain the coordinates X and Y by calculating λ using the relation:

$$\lambda = \frac{Z - C}{z_3}; (z_1, z_2, z_3)^T = R^{-1} K^{-1} p \tag{5.18}$$

The back-projection operation is important for depth estimation and image rendering, which will be extensively addressed later in this thesis. For depth estimation, this would mean that an assumption is made for the value of Z and the corresponding $3D$ point is calculated. With an iterative procedure, an appropriate depth value is selected from a set of assumed depth candidates.

5.1.3 Coordinate system conversion

Sometimes, coordinate systems need to be converted to obtain a more efficient computation procedure. Let us now propose two methods that transform the projection

matrix, so that new coordinate systems can be employed. We will also provide applications of those methods.

A. Changing the image coordinate system

The definition of the coordinate system in 3D image processing is not uniformly chosen. A right-handed coordinate system is usually employed in literature. Therefore, we outline a method for converting the image coordinate system.

Typically, pixel coordinates are defined such that the origin of the $2D$ image coordinate system is located at the top left of the image. In this case, the x and y axis point horizontally to the right and vertically downward, respectively (convention 1). However, an alternative convention is to locate the origin of the image coordinate system at the bottom left, with the y image axis pointing vertically upwards. To transform the image coordinate system, it is necessary to flip the y image axis and translate the origin along the y image axis. This can be performed using the matrix denoted B_1 (see Equation.5.19). Additionally, one can distinguish two possible conventions for defining the orientation of the $3D$ world axis: either a left-handed or a right-handed coordinate system can be adopted. The conversion of a left-handed to a right-handed coordinate system can be performed by flipping the Y world axis 1 (matrix B_2). By concatenating the two conversion matrices B_1 and B_2 with the original projection matrix, one can obtain the converted projection matrix:

$$\lambda p = \underbrace{\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & h-1 \\ 0 & 0 & 1 \end{bmatrix}}_{B_1} [K|0_3] \begin{bmatrix} R & -RC \\ 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{B_2} P}_{\text{Converted Projection Matrix}} \tag{5.19}$$

where h corresponds to the height of the image. The obtained converted projection matrix is then defined in an image coordinate system using the convention 1 notation and in a right-handed world coordinate system. Finally, it should be noted that the conversion of the image coordinate system is achieved by modifying the intrinsic parameters while the conversion of the world coordinate system is made by transforming the extrinsic parameters. This is the reason why conversion matrices

B_1 and B_2 are placed as left and right terms in Equation.5.19.

B. Changing the world coordinate system

A conversion is used for re-specifying depth images into a new world coordinate system. This conversion involves the calculation of the position of a $3D$ point specified in another camera coordinate system and the projection of this $3D$ point onto the other image plane. The modification of the location and orientation of the world coordinate system is performed in a way similar to the above-described method.

The modification of the world coordinate system involves the simultaneous conversion of the projection matrix and the coordinates of the $3D$ point. Considering a $3D$ world point P and a camera defined with a projection matrix with intrinsic and extrinsic parameters K, R and C, the coordinate-system conversion can be carried out in two steps. First, specify the projection matrix in a new world coordinate system, where only the position and orientation of the camera, i.e., extrinsic parameters, should be modified. The extrinsic parameters are converted using the position C_n and orientation R_n of the new coordinate system defined, with respect to the original coordinate system. Second, specify the position of a $3D$ point P in the new coordinate system. The coordinate-system conversion can be written as:

$$\lambda p = [K|0_3] \underbrace{\begin{bmatrix} R & -RC \\ 0_3^T & 1 \end{bmatrix} \begin{bmatrix} R_n^T & C_n \\ 0_3^T & 1 \end{bmatrix}}_{\text{Converted Extrinsic Parameteres}} \underbrace{\begin{bmatrix} R_n & -R_nC_n \\ 0_3^T & 1 \end{bmatrix}}_{\text{Converted 3D Position}} P \tag{5.20}$$

where p represents the projected pixel position and the all-zero element vector is denoted by 0_3.

5.2 Camera calibration

Camera calibration involves the estimation of both extrinsic and intrinsic camera parameters. This section does not present a new calibration algorithm, but instead shortly describes a parameter-estimation technique for calibrating a multi-view camera setup. To compute the camera parameters, a practical technique [179] based on a calibration rig is employed. As for correcting the lens distortion, the estimation of camera parameters is based on a calibration rig with known geometry, such as

a checkerboard pattern. Using various perspective views of the checkerboard, the algorithm estimates the position, orientation and internal parameters of the camera. The process of estimating all camera parameters is known as strong calibration, which will be addressed in the sequel.

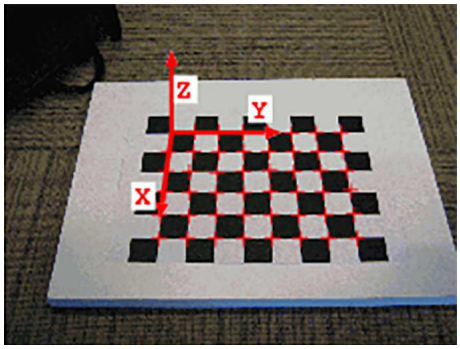


Figure 5.8: A planar checkerboard pattern defines a 3D world coordinate system, and additionally enables the detection of 3D feature points.

5.2.1 Camera parameters estimation

Let us consider a planar checkerboard that defines a right-handed $3D$ world coordinate system (see Figure.5.8). The first stage of the camera calibration procedure is to establish correspondences between $2D$ points in the image and $3D$ points on the checkerboard, i.e., so-called point-correspondences. In practice, a robust extraction of point-correspondences can be performed by exploiting the topological structure of the checkerboard pattern [180].

Because each $3D$ feature point belongs to the board plane $Z = 0$, the projection of each $2D$ point onto the image plane can be written as:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = [K|0_3] \begin{bmatrix} R & -RC \\ 0_3^T & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z = 0 \\ 1 \end{pmatrix} \tag{5.21}$$

which can be reformulated to:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \underbrace{K \begin{bmatrix} r_1 & r_2 & t \end{bmatrix}}_{\text{Homography Transform}} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{5.22}$$

where r_i corresponds to the i^{th} column of the rotation matrix R and $t = -RC$. Because the matrix product $K[r_1, r_2, t]$ is a 3×3 matrix, it corresponds to a so-called *planar homography transform*, which is typically denoted as a matrix H. A planar homography transform is a non-singular linear relation between two planes [181]. In our case, the homography transform defines a linear mapping of points between the planar checkerboard (3D position) and the image plane (pixel position). The calculation of the camera parameters requires the estimation of the homography transform H:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}}_{\text{Homography H}} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{5.23}$$

A. Homography transform estimation

The homography H can be estimated by detecting point-correspondences $p_i \leftrightarrow P'_i$ with $p_i = (x_i, y_i, 1)^T$ being the pixel position and $P'_i = (X_i, Y_i, 1)^T$ being the world coordinates of the corresponding feature point of index i on the checkerboard. Using the previously introduced notation, Equation.5.23 may be written as:

$$\lambda p_i = H P_i \tag{5.24}$$

In order to eliminate the scaling factor, one can calculate the cross product of each term of Equation.5.24, leading to:

$$p_i \times (\lambda p_i) = p_i \times (p_i \times H {P'}_i) \tag{5.25}$$

and since $p_i \times p_i = 0_3$, Equation.5.25 may be written as:

$$p_i \times H {P'}_i = 0_3 \tag{5.26}$$

Writing the matrix product HP'_i as:

$$HP'_i = \begin{bmatrix} h^{1T}P'_i \\ h^{2T}P'_i \\ h^{3T}P'_i \end{bmatrix} \tag{5.27}$$

with h^{mT} being the transpose of the m^{th} row of H, we rework Equation.5.26 into:

$$p_i \times HP'_i = \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \times \begin{bmatrix} h^{1T}P'_i \\ h^{2T}P'_i \\ h^{3T}P'_i \end{bmatrix} = \begin{bmatrix} y_i h^{3T}P'_i - h^{2T}P'_i \\ h^{1T}P'_i - x_i h^{3T}P'_i \\ x_i h^{2T}P'_i - y_i h^{1T}P'_i \end{bmatrix} = 0_3 \tag{5.28}$$

Since Equation.5.28 is linear in h^{mT} and noting that $h^{mT}P'_i = {P'_i}^T h^m$, Equation.5.28 may be reformulated as:

$$\begin{bmatrix} 0_3^T & -{P'_i}^T & y_i{P'_i}^T \\ {P'_i}^T & 0_3^T & -x_i{P'_i}^T \\ -y_i{P'_i}^T & x_i{P'_i}^T & 0_3^T \end{bmatrix} \begin{bmatrix} h^1 \\ h^2 \\ h^3 \end{bmatrix} = 0_9 \tag{5.29}$$

Note that the rows of the previous matrix are not linearly independent: the third row is the sum of $-x_i$ times the first row and $-y_i$ times the second row. Thus, for each point-correspondence, Equation.5.29 provides two linearly independent equations. The two first rows are typically used for solving H. Because the homography transform is written using homogeneous coordinates, the homography H is defined using 8 parameters plus a free 9^{th} homogeneous scaling factor. Therefore, at least 4 point-correspondences providing 8 equations are required to compute the homography. Practically, a larger number of correspondences is employed so that an over-determined linear system is obtained. By rewriting H in a vector form as $h = [h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}, h_{33}]^T$, n pairs of point-correspondences enable the construction of a $2n \times 9$ linear system, which is expressed by (i corresponds to the index of the feature point, hence $i = 1$ for first two rows, $i = 2$ for the second

two rows, etc.):

$$\underbrace{\begin{bmatrix} 0 & 0 & 0 & -X_1 & -Y_1 & -1 & y_1X_1 & y_1X_1 & y \\ X_1 & Y_1 & 1 & 0 & 0 & 0 & -x_1X_1 & -x_1Y_1 & -x_1 \\ 0 & 0 & 0 & -X_2 & -Y_2 & -1 & y_2X_2 & y_2X_2 & y_2 \\ X_2 & Y_2 & 1 & 0 & 0 & 0 & -x_2X_2 & -x_2Y_2 & -x_2 \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & -X_n & -Y_n & -1 & y_nX_n & y_nX_n & y_n \\ X_n & Y_n & 1 & 0 & 0 & 0 & -x_nX_n & -x_nY_n & -x_n \end{bmatrix}}_{C} \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{pmatrix} = 0_9 \quad (5.30)$$

Solving this linear system involves the calculation of a Singular Value Decomposition (SVD). Such an SVD corresponds to reworking the matrix to the form of the matrix product $C = UDV^T$, where the solution h corresponds to the last column of the matrix V . To avoid numerical instabilities, the coordinates of point-correspondences should be normalized. This technique is known as the normalized Direct Linear Transformation (DLT) algorithm [181]. It should be noted that a non-linear refinement of the solution may be performed afterwards. However, such a non-linear refinement does not yield a significant improvement and moreover, it involves the calculation of a Jacobian matrix, required for the non-linear Levenberg-Marquardt optimization.

B. Calculation of intrinsic parameters

Assuming that the homography transform H is calculated, the intrinsic parameters can be recovered using the a-priori knowledge that r_1 and r_2 are orthonormal. Denoting the homography transform as $H = [h_1\ h_2\ h_3]$, it can be derived from Equation.5.22 that $[h_1,\ h_2] = \lambda' K \cdot [r_1, r_2]$ which is equivalent to $K^{-1} \cdot [h_1, h_2] = \lambda' [r_1, r_2]$ where λ' is a homogeneous factor. Assuming that the vectors r_1, r_2 of the rotation matrix R are orthonormal, then the vectors r_1 and r_2 are perpendicular to each other and have equal norm, so that:

$$\begin{aligned} &(h_1^T K^{-T} \cdot K^{-1} h_2) = 0 \\ &h_1^T K^{-T} \cdot K^{-1} h_1 = h_2^T K^{-T} \cdot K^{-1} h_2 \end{aligned} \quad (5.31)$$

The internal term $K^{-T}K^{-1}$ appears regularly, which for further use is defined as B and can be written as:

$$B = K^{-T}K^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha^2} & -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma o_y - \beta o_x}{\alpha^2\beta} \\ -\frac{\gamma}{\alpha^2\beta} & \frac{\gamma^2+\alpha^2}{\alpha^2\beta^2} & \frac{(-\gamma^2-\alpha^2)o_y+\beta\gamma o_x}{\alpha^2\beta^2} \\ \frac{\gamma o_y - \beta o_x}{\alpha^2\beta} & \frac{(-\gamma^2-\alpha^2)o_y+\beta\gamma o_x}{\alpha^2\beta^2} & \frac{(\gamma^2+\alpha^2)o_y^2-2\beta\gamma o_x o_y+\beta^2 o_y^2+\alpha^2\beta^2}{\alpha^2\beta^2} \end{bmatrix} \tag{5.32}$$

This matrix covers the intrinsic parameters of the camera, such as the focal length $\alpha = f$, aspect-ratio related parameter $\beta = \eta f$ and skew $\gamma = \tau$. This description corresponds to the notation adopted in the literature [179]. By writing B in a vector form and omitting the double terms from the symmetry in B, we find a reduced vector b with only 6 terms, thus $b = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^T$. Referring to Equation.5.31, and starting with the bottom equation and generalizing it with the difference in index in the top equation, we can write the general expression of ${h_k}^T B h_l$ as:

$${h_k}^T B h_l = v_{kl}^T b \tag{5.33}$$

with:

$$v_{kl}^T = \begin{bmatrix} h_{k1}h_{l1} & h_{k1}h_{l2} + h_{k2}h_{l1} & h_{k2}h_{l2} & h_{k3}h_{l1} + h_{k1}h_{l3} & h_{k3}h_{l2} + h_{k2}h_{l3} & h_{k3}h_{l3} \end{bmatrix} \tag{5.34}$$

and h_k being the k^{th} column vector of H with $h_k = [h_{k1}, h_{k2}, h_{k3}]^T$. For each homography or image of the checkerboard, Equation.5.31 can be written as:

$$\begin{bmatrix} v_{12}^T \\ (v_{11} - v_{22})^T \end{bmatrix} b = 0 \tag{5.35}$$

Note that the vector b which summarizes the intrinsic parameters, is a 6-element vector so that 6 equations are necessary to recover all camera parameters. Therefore, since each homography provides 2 linear equations, at least 3 homographies or captured images are sufficient. Assuming that n homography transforms are known,

the linear system composed of n instances of Equation.5.35 can be written as:

$$Vb = 0 \tag{5.36}$$

with V referring to a $2n \times 6$ matrix. This linear system can be solved by employing the standard technique of Singular Value Decomposition. Using the computed solution vector b, the intrinsic parameters can be derived [179] as follows. The computed vector $b = [B_{11}, B_{12}, B_{22}, B_{13}, B_{23}, B_{33}]^T$ is expanded into the symmetric matrix B, by inserting again the omitted symmetric matrix elements. Rewriting the matrix B as $B = \lambda K^{-T} K^{-1}$ (Equation.5.32), we finally find the intrinsic parameters as:

$$\begin{aligned} o_y &= \tfrac{(B_{12}B_{13} - B_{11}B_{23})}{B_{11}B_{22} - B_{12}^2} \\ \lambda &= B_{33} - \tfrac{B_{13}^2 + o_y(B_{12}B_{13} - B_{11}B_{23})}{B_{11}} \\ \alpha &= \sqrt{\tfrac{\lambda}{B_{11}}} \\ \beta &= \sqrt{\tfrac{\lambda B_{11}}{B_{11}B_{22} - B_{12}^2}} \\ \gamma &= \tfrac{-B_{12}\alpha^2\beta}{\lambda} \\ o_x &= \tfrac{\gamma o_y}{\beta} - \tfrac{B_{13}\alpha^2}{\lambda} \end{aligned} \tag{5.37}$$

C. Calculation of extrinsic parameters

The position and orientation of the camera can be recovered from Equation.5.22 and written as:

$$\begin{aligned} r_1 &= \lambda_1 K^{-1} h_1 \\ r_2 &= \lambda_2 K^{-1} h_2 \\ t &= -RC = \lambda_3 K^{-1} h_3 \end{aligned} \tag{5.38}$$

Given the vectors r_1 and r_2 and bearing in mind that the rotation matrix is orthogonal, the third vector r_3 to complete the matrix R is found by:

$$r_3 = r_1 \times r_2 \tag{5.39}$$

where $\times$ denotes a cross product. The scaling parameters are calculated by $\lambda_1 = 1/ \parallel K^{-1}h_1 \parallel$ and $\lambda_2 = 1/ \parallel K^{-1}h_2 \parallel$ and $\lambda_3 = (\lambda_1 + \lambda_2)/2$. Theoretically, we have $\lambda_1 = \lambda_2$. However, due to inaccuracies in the feature-point estimation procedure, both terms may differ and have to be treated separately.

D. Non-linear refinement of the camera parameters

To refine the obtained camera parameters, it is possible to perform a non-linear minimization of a projection function. This function is projecting the $3D$ points onto the image plane and accumulating the differences between corresponding points. The function is specified by:

$$\sum_{j}^{N}\sum_{i}^{n}\left\|p_{ij}-\left([K|0_3]\begin{bmatrix} R_j & -R_jC_j \\ 0 & 1 \end{bmatrix}P_{ij}\right)\right\|^2 \tag{5.40}$$

where j denotes the image index and i corresponds to the point-correspondence index.

E. Summary of camera calibration steps

The camera calibration algorithm can be summarized as follows.

1. Capture N (at least 3) images with the checkerboard pattern and estimate point-correspondences.
2. Estimate and correct the radial lens distortion.
3. For each captured image, calculate the N homography transforms.
4. Using the N homography transforms, calculate the intrinsic and extrinsic parameters.
5. Refine the calculated camera parameters as explained in Section D above.

As opposed to the standard technique, we perform the correction of the non-linear radial lens distortion prior to the camera calibration procedure. This slight modification yields more accurate results for obtaining the homographies H_i and, afterwards, more accurate calibration parameters.

5.3 Two-view geometry

In the previous section, we have described the geometry of a single camera. We now examine the case of two camera views capturing the same scene from two different viewpoints. Given two images, we are interested in estimating the $3D$ structure of

the scene. The estimation of coordinates of a $3D$ point P can be performed in two steps. First, given a selected pixel p_1 in the image I_1, the position of the corresponding pixel p_2 in image I_2 is estimated. Similarly to the previous section, a pair of corresponding points p_1 and p_2 is called a point-correspondence. This correspondence in points p_1 and p_2 comes from the projection of the same point P onto both images I_1 and I_2. Second, the position of P is calculated by triangulation of the two corresponding points, using the geometry of the two cameras. The geometry of the two cameras relates to the respective position and orientation and internal geometry of each individual camera. The underlying geometry that describes the relationship between both cameras is known as the epipolar geometry. The estimation process of the epipolar geometry is known as weak calibration, as opposed to the strong calibration mentioned earlier in this chapter. The epipolar geometry addresses (among others) the following two aspects.

- Geometry of point-correspondence: considering a point in an image, the epipolar geometry provides a constraint on the position of the corresponding point.
- Scene geometry: given point-correspondences and the epipolar geometry of both cameras, a description of the scene structure can be recovered.

5.3.1 Epipolar geometry

Let us now describe the geometry of two images and define their mutual relation. Consider a $3D$ point P that is projected through the camera centers C_1 and C_2 onto two images at pixel positions p_1 and p_2, respectively (see Figure.5.9). Obviously, the $3D$ points P, C_1, C_2 and the projected points p_1, p_2 are all located within one common plane. This common plane denoted π is known as the epipolar plane. The epipolar plane is fully determined by the back-projected ray going through C_1 and p_1 and the camera center C_2. The property that the previously specified points belong to the epipolar plane provides a constraint for searching point-correspondences. More specifically, considering the image point p_1, a point p_2 lies on the intersection of the plane π with the second image plane (denoted within I_2 in Figure.5.9). The intersection of both planes corresponds to a line known as the epipolar line. Therefore, the search of point-correspondences can be limited to a search along the epipolar

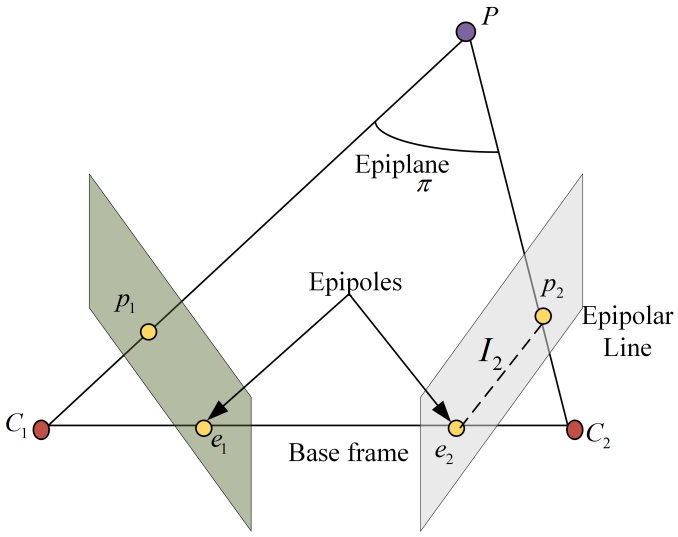


Figure 5.9: Epipolar geometry. The epipolar plane is defined by the point P and the two camera centers C_1 and C_2. Two cameras and image planes with an indication of the terminology of epipolar geometry.

line instead of an exhaustive search in the image. Additionally, it is interesting to note that this constraint is independent of the scene structure, but instead, uniquely relies on the epipolar geometry. The epipolar geometry can be described using a 3×3 rank-2 matrix, called the fundamental matrix F, which is defined by $l_2 = Fp_1$.

We now introduce some terminology related to the epipolar geometry employed further in this thesis.

- The *epipolar plane* is the plane defined by a $3D$ point and the two cameras centers.
- The *epipolar line* is the line determined by the intersection of the image plane with the epipolar plane.
- The *baseline* is the line going through the two cameras centers.
- The epipole is the image-point determined by the intersection of the image plane with the baseline. Also, the epipole corresponds to the projection of

the first camera center (say C_1) onto the second image plane (say I_2), or vice versa.

As previously highlighted, three-dimensional structural information can be extracted from two images by exploiting the epipolar geometry. It was shown that the three-dimensional structure can be extracted by determining the correspondences in the two views and that the point-corresponden-ces can be searched along the epipolar line only. To simplify the search, images are typically captured such that all epipolar lines are parallel and horizontal. In this case, the search of point-correspondences can be performed along the horizontal raster lines of both images. However, it is difficult to accurately align and orient the two cameras such that epipolar lines are parallel and horizontal. Instead, an alternative approach is to capture two views (without alignment and orientation constraints) and transform both images to synthesize two novel views with parallel and horizontal epipolar lines. This procedure is called image rectification which will be described in the next section.

5.3.2 Image rectification

Image rectification is the process of transforming two images I_1 and I_2 such that their epipolar lines are horizontal and parallel. This procedure is particularly useful for depth-estimation algorithms because the search of point-correspondences can be performed along horizontal raster image lines. Practically, the image-rectification operation corresponds to a virtual rotation of two cameras, so that they would become aligned. As illustrated by Figure.5.10, an image-rectification technique [182] consists of synthesizing a common image plane I' and re-projecting the two images I_1 and I_2 onto this synthetic plane. We now derive the transform between the original image I_1 and rectified image I'_1. Let us consider a pixel p_1 and its projections on the rectified image p'_1 in I_1 and I'_1, respectively (illustrated by Figure.5.11). Without loss of generality, it is assumed that the camera is located at the origin of the world coordinate system.

The projection of a $3D$ point $(X, Y, Z)^T$ onto the original and rectified images

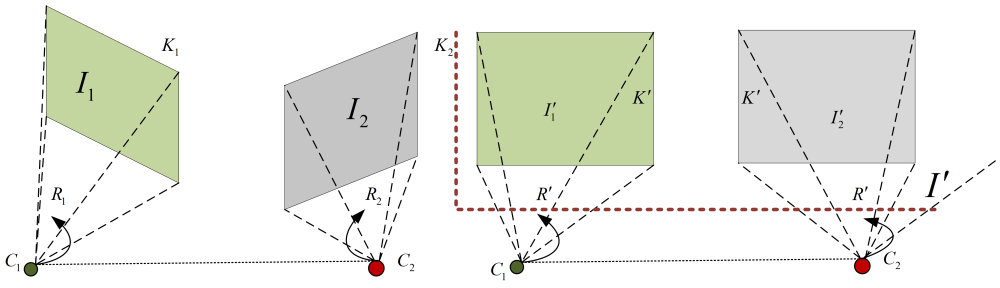


Figure 5.10: The image-rectification operation re-projects two images I_1 and I_2 onto a common synthetic image plane I'.

can be written as:

$$\begin{aligned} \lambda_1 p_1 &= K_1 R_1 \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \\ \lambda'_1 p'_1 &= K'_1 R'_1 \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \end{aligned} \tag{5.41}$$

with R_1, K_1 and R', K' being the original and (virtual) rectification matrices, respectively. By recombining both previous relations, we obtain the transformation between the original and rectified image planes defined as:

$$\frac{\lambda'_1}{\lambda_1} p'_1 = \underbrace{K'_1 R'_1 R_1^{-1} K_1^{-1}}_{H_1} p_1 \tag{5.42}$$

Similarly, the rectification of image I_2 is performed using the relation:

$$\frac{\lambda'_2}{\lambda_2} p'_2 = \underbrace{K'_2 R'_2 R_2^{-1} K_2^{-1}}_{H_2} p_2 \tag{5.43}$$

When observing Equation.5.42, it can be noted that the four matrices can be combined into a 3×3 matrix that corresponds to a homography transform. We now provide details on the computation of the rotation and projection matrix R' and K'.

A. Calculation of matrix R'

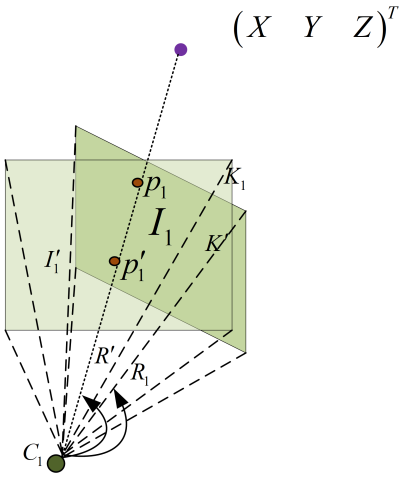


Figure 5.11: A 3D point $(X, Y, Z)^T$ is projected onto the original and rectified image planes I_1 and I'_1 at pixel position p1 and p'_1.

A single rotation matrix R' can rotate the two cameras towards the same direction. The rotation matrix $R' = (r'^x, r'^y, r'^z)^T$ can be calculated as follows:

- The row r'^x can be defined parallel to the baseline going through the two cameras centers C_1 and C_2, leading to $r'^x = \frac{C1-C2}{\|C1-C2\|}$.
- The row r'^y can be chosen as $r'^y = r^z \times r'^x$ where r^z is the first row vector of the rotation matrix R_1. In this case, the new axis r'^y is perpendicular to both the new r'^x and the old r^z of matrix R_1.
- The row r'^z is defined orthogonal to r'^x and r'^y such that $r'^z = r'^x \times r'^y$.

Note that each vector is normalized by $r^k := r^k/||r^k||$, where $||r_k||$ represents the $L2$-Norm of the vector and $k \in \{x, y, z\}$.

B. Calculation of matrix K'

The intrinsic parameters of K' must be equal for both image-rectification operations and can be defined arbitrarily, e.g., $K' = \frac{K1+K2}{2}$.

C. Calculation of the bounding box

Basically, Equation.5.42 means that the image rectification corresponds to a homography transform. Because a homography transforms rectangles into quadrilaterals

of arbitrary size, the size of the output image needs to be calculated. The bounding box calculation needs to be carried out for both rectified images and the largest bounding box should be selected as the common bounding box. The upper-left and bottom-right corners of the two rectified images are denoted (minx,miny) and (maxx,maxy) (see Figure.5.12). Subsequently, the width and height of the output image can be written as $w' = max_x - min_x$ and $h' = max_y - min_y$, respectively. Additionally, as illustrated at the corner-points in Figure.5.12, some points with positive coordinates may be projected at negative pixel positions. Consequently, we

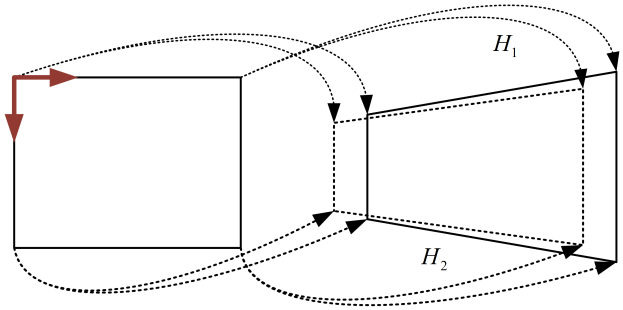


Figure 5.12: Bounding box calculation of the rectified images derived from a planar homography transform.

modify the homography transforms H_1 and H_2 to obtain new homographies H'_1 and H'_2 that map all input pixels to positive coordinates, where H'_i, for i is 1 or 2, can be defined by:

$$H'_i = TH_i; T = \begin{bmatrix} 1 & 0 & -\min_x \\ 0 & 1 & -\min_y \\ 0 & 0 & 1 \end{bmatrix} \tag{5.44}$$

The matrix T corresponds to a translation matrix expressed in homogeneous coordinates.

D. Algorithm summary

The rectification homographies H'_1 and H'_2 can be defined as:

$$\begin{aligned} H'_1 &= TK'R'R_1^{-1}K_1^{-1} \\ H'_2 &= TK'R'R_2^{-1}K_2^{-1} \end{aligned} \tag{5.45}$$

with T, R', K' defined as in the previous paragraphs. As an example, Figure.5.13 depicts two rectified images with two superimposed horizontal epipolar lines.

Figure 5.13: Bounding box calculation of the rectified images derived from a planar homography transform.

5.4 Optical Flow

Optical flow or optic flow, shown in figure.5.14 is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer and a scene [183,184]. The concept of optical flow was introduced by the American psychologist James J. Gibson in the 1940s to describe the visual stimulus provided to animals moving through the world [185]. Gibson stressed the importance of optic flow for affordance perception, the ability to discern possibilities for action within the environment. Followers of Gibson and his ecological approach to psychology have further demonstrated the role of the optical flow stimulus for the perception of movement by the observer in the world; perception of the shape, distance and movement of objects in the world; and the control of locomotion [186]. The term optical flow is also used by roboticists, encompassing related techniques from image processing and control of navigation including motion detection, object segmentation, time-to-contact information, focus of expansion calculations, luminance, motion compensated encoding, and stereo disparity measurement [187].

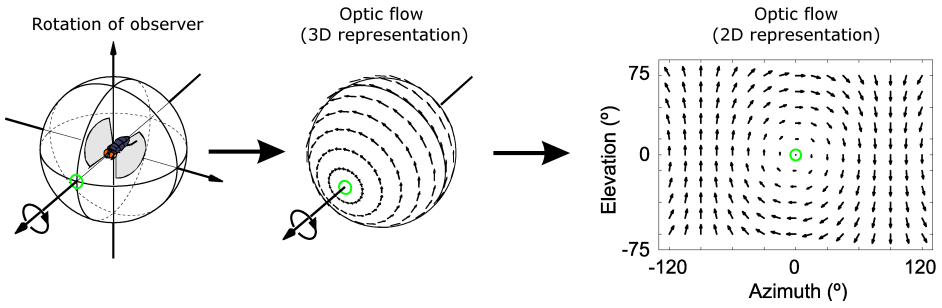


Figure 5.14: The optic flow experienced by a rotating observer (in this case a fly). The direction and magnitude of optic flow at each location is represented by the direction and length of each arrow.

5.4.1 Estimation

Sequences of ordered images allow the estimation of motion as either instantaneous image velocities or discrete image displacements [188]. Fleet and Weiss [189] provide a tutorial introduction to gradient based optical flow. John L. Barron, David J. Fleet, and Steven Beauchemin provide a performance analysis of a number of optical flow techniques. It emphasizes the accuracy and density of measurements [190].

The optical flow methods try to calculate the motion between two image frames which are taken at times t and $t + \Delta t$ at every voxel position. These methods are called differential since they are based on local Taylor series approximations of the image signal; that is, they use partial derivatives with respect to the spatial and temporal coordinates.

For a $2D+t$ dimensional case ($3D$ or $n-D$ cases are similar) a voxel at location (x, y, t) with intensity $I(x, y, t)$ will have moved by Δx, Δy and Δt between the two image frames, and the following brightness constancy constraint can be given:

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \tag{5.46}$$

Assuming the movement to be small, the image constraint at $I(x, y, t)$ with Taylor

series can be developed to get:

$$I(x+\Delta x, y+\Delta y, t+\Delta t) = I(x,y,t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t + \text{H.O.T} \quad (5.47)$$

From these equations it follows that:

$$\frac{\partial I}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial I}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial I}{\partial t}\frac{\Delta t}{\Delta t} = 0 \quad (5.48)$$

which results in:

$$\frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t} = 0 \quad (5.49)$$

where V_x, V_y are the x and y components of the velocity or optical flow of $I(x,y,t)$ and $\frac{\partial I}{\partial x}$, $\frac{\partial I}{\partial y}$ and $\frac{\partial I}{\partial t}$ are the derivatives of the image at (x,y,t) in the corresponding directions. I_x, I_y and I_t can be written for the derivatives in the following. Thus:

$$I_x V_x + I_y V_y = -I_t \text{ or } \nabla I^T \cdot \vec{V} = -I_t \quad (5.50)$$

This is an equation in two unknowns and cannot be solved as such. This is known as the aperture problem of the optical flow algorithms. To find the optical flow another set of equations is needed, given by some additional constraint. All optical flow methods introduce additional conditions for estimating the actual flow.

5.4.2 Methods for determination

In literature over years many methods have been developed for estimation of optical flow. Because of our special usage (Industrial Visual Servoing) we have listed some important methods as:

- ***Phase correlation:*** inverse of normalized cross-power spectrum.
- ***Block-based methods:*** minimizing sum of squared differences or sum of absolute differences, or maximizing normalized cross-correlation.
- ***Differential methods of estimating optical flow:*** based on partial derivatives of the image signal and/or the sought flow field and higher-order partial derivatives, such as:

  - ***Lucas–Kanade method:*** regarding image patches and an affine model for the flow field.
  - ***Horn–Schunck method:*** optimizing a functional based on residuals from the brightness constancy constraint, and a particular regularization term expressing the expected smoothness of the flow field.
  - ***Buxton–Buxton method:*** based on a model of the motion of edges in image sequences.
  - ***Black–Jepson method:*** coarse optical flow via correlation.
  - ***General variational methods:*** a range of modifications/extensions of Horn–Schunck, using other data terms and other smoothness terms.

- ***Discrete optimization methods:*** the search space is quantized, and then image matching is addressed through label assignment at every pixel, such that the corresponding deformation minimizes the distance between the source and the target image. The optimal solution is often recovered through Max-flow min-cut theorem algorithms, linear programming or belief propagation methods.

Many of these, in addition to the current state-of-the-art algorithms are evaluated on the Middlebury Benchmark Dataset [191].

Motion estimation and video compression have developed as a major aspect of optical flow research. While the optical flow field is superficially similar to a dense motion field derived from the techniques of motion estimation, optical flow is the study of not only the determination of the optical flow field itself, but also of its use in estimating the three-dimensional nature and structure of the scene, as well as the 3D motion of objects and the observer relative to the scene, most of them using the Image Jacobian.

Optical flow was used by robotics researchers in many areas such as: object detection and tracking, image dominant plane extraction, movement detection, robot navigation and visual odometry [187]. Optical flow information has been recognized as being useful for controlling micro air vehicles [192].

The application of optical flow includes the problem of inferring not only the motion of the observer and objects in the scene, but also the structure of objects

and the environment. Since awareness of motion and the generation of mental maps of the structure of our environment are critical components of animal (and human) vision, the conversion of this innate ability to a computer capability is similarly crucial in the field of machine vision. Consider a five-frame clip of a ball moving

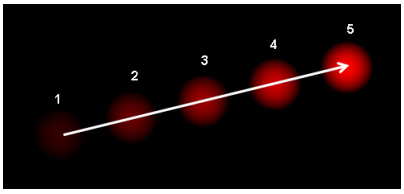


Figure 5.15: The optical flow vector of a moving object in a video sequence.

from the bottom left of a field of vision in figure.5.15, to the top right. Motion estimation techniques can determine that on a two dimensional plane the ball is moving up and to the right and vectors describing this motion can be extracted from the sequence of frames. For the purposes of video compression (e.g., MPEG), the sequence is now described as well as it needs to be. However, in the field of machine vision, the question of whether the ball is moving to the right or if the observer is moving to the left is unknowable yet critical information. Not even if a static, patterned background were present in the five frames, could we confidently state that the ball was moving to the right, because the pattern might have an infinite distance to the observer.

5.4.3 Optical flow sensor

An optical flow sensor is a vision sensor capable of measuring optical flow or visual motion and outputting a measurement based on optical flow. Various configurations of optical flow sensors exist. One configuration is an image sensor chip connected to a processor programmed to run an optical flow algorithm. Another configuration uses a vision chip, which is an integrated circuit having both the image sensor and the processor on the same die, allowing for a compact implementation.An example of this is a generic optical mouse sensor used in an optical mouse. In some cases

the processing circuitry may be implemented using analog or mixed-signal circuits to enable fast optical flow computation using minimal current consumption.

One area of contemporary research is the use of neuromorphic engineering techniques to implement circuits that respond to optical flow, and thus may be appropriate for use in an optical flow sensor. Such circuits may draw inspiration from biological neural circuitry that similarly responds to optical flow. Optical flow sensors are used extensively in computer optical mice, as the main sensing component for measuring the motion of the mouse across a surface. Optical flow sensors are also being used in robotics applications, primarily where there is a need to measure visual motion or relative motion between the robot and other objects in the vicinity of the robot. The use of optical flow sensors in unmanned aerial vehicles (UAVs), for stability and obstacle avoidance, is also an area of current research.

5.5 Lucas–Kanade method

In computer vision, the Lucas–Kanade method is a widely used differential method for optical flow estimation developed by Bruce D. Lucas and Takeo Kanade. It assumes that the flow is essentially constant in a local neighbourhood of the pixel under consideration, and solves the basic optical flow equations for all the pixels in that neighbourhood, by the least squares criterion [193, 194].

By combining information from several nearby pixels, the Lucas–Kanade method can often resolve the inherent ambiguity of the optical flow equation. It is also less sensitive to image noise than point-wise methods. On the other hand, since it is a purely local method, it cannot provide flow information in the interior of uniform regions of the image.

5.5.1 Concept

The Lucas–Kanade method assumes that the displacement of the image contents between two nearby instants (frames) is small and approximately constant within a neighborhood of the point p under consideration. Thus the optical flow equation can be assumed to hold for all pixels within a window centered at p. Namely, the

local image flow (velocity) vector (V_x, V_y) must satisfy:

$$\begin{aligned} I_x(q_1)V_x + I_y(q_1)V_y &= -I_t(q_1) \\ I_x(q_2)V_x + I_y(q_2)V_y &= -I_t(q_2) \\ &\vdots \\ I_x(q_n)V_x + I_y(q_n)V_y &= -I_t(q_n) \end{aligned} \tag{5.51}$$

where $q_1, q_2, \ldots, q_n$ are the pixels inside the window, and $I_x(q_i), I_y(q_i), I_t(q_i)$ are the partial derivatives of the image I with respect to position x, y and time t, evaluated at the point q_i and at the current time. These equations can be written in matrix form $Av = b$, where:

$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1) \\ I_x(q_2) & I_y(q_2) \\ \vdots & \vdots \\ I_x(q_n) & I_y(q_n) \end{bmatrix} \qquad v = \begin{bmatrix} V_x \\ V_y \end{bmatrix} \qquad b = \begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ \vdots \\ -I_t(q_n) \end{bmatrix} \tag{5.52}$$

This system has more equations than unknowns and thus it is usually over-determined. The Lucas–Kanade method obtains a compromise solution by the least squares principle. Namely, it solves the 2×2 system:

$$A^T A v = A^T b \text{ or } \mathrm{v} = (A^T A)^{-1} A^T b \mathrm{v} = (A^T A)^{-1} A^T b \tag{5.53}$$

That is, it computes:

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i) I_y(q_i) \\ \sum_i I_y(q_i) I_x(q_i) & \sum_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i) I_t(q_i) \\ -\sum_i I_y(q_i) I_t(q_i) \end{bmatrix} \tag{5.54}$$

where the central matrix in the equation is an Inverse matrix. The sums are running from $i = 1$ to n. The matrix $A^T A$ is often called the structure tensor of the image at the point p.

5.5.2 Weighted window

The plain least squares solution above gives the same importance to all n pixels q_i in the window. In practice it is usually better to give more weight to the pixels that are closer to the central pixel p. For that, one uses the weighted version of the least squares equation:

$$A^T W A v = A^T W b \tag{5.55}$$

where W is an $n \times n$ diagonal matrix containing the weights $W_{ii} = w_i$ to be assigned to the equation of pixel q_i. That is, it computes:

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i w_i I_x(q_i)^2 & \sum_i w_i I_x(q_i) I_y(q_i) \\ \sum_i w_i I_x(q_i) I_y(q_i) & \sum_i w_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i w_i I_x(q_i) I_t(q_i) \\ -\sum_i w_i I_y(q_i) I_t(q_i) \end{bmatrix} \tag{5.56}$$

The weight w_i is usually set to a Gaussian function of the distance between q_i and p.

5.5.3 Use conditions and techniques

In order for equation $A^T A v = A^T b$ to be solvable, $A^T A$ should be invertible, or $A^T A$'s eigenvalues satisfy $\lambda_1 \geq \lambda_2 > 0$. To avoid noise issue, usually λ_2 is required to not be too small. Also, if λ_1/λ_2 is too large, this means that the point p is on an edge, and this method suffers from the aperture problem. So for this method to work properly, the condition is that λ_1 and λ_2 are large enough and have similar magnitude. This condition is also the one for Corner detection. This observation shows that one can easily tell which pixel is suitable for the Lucas–Kanade method to work on by inspecting a single image.

One main assumption for this method is that the motion is small (less than 1 pixel between two images for example). If the motion is large and violates this assumption, one technique is to reduce the resolution of images first and then apply the Lucas-Kanade method.

5.5.4 Improvements and extensions

The least-squares approach implicitly assumes that the errors in the image data have a Gaussian distribution with zero mean. If one expects the window to contain a certain percentage of "outliers" (grossly wrong data values, that do not follow the "ordinary" Gaussian error distribution), one may use statistical analysis to detect them, and reduce their weight accordingly.

The Lucas–Kanade method per se can be used only when the image flow vector V_x, V_y between the two frames is small enough for the differential equation of the optical flow to hold, which is often less than the pixel spacing. When the flow vector may exceed this limit, such as in stereo matching or warped document registration, the Lucas–Kanade method may still be used to refine some coarse estimate of the same, obtained by other means; for example, by extrapolating the flow vectors computed for previous frames, or by running the Lucas-Kanade algorithm on reduced-scale versions of the images. Indeed, the latter method is the basis of the popular Kanade-Lucas-Tomasi (KLT) feature matching algorithm.

A similar technique can be used to compute differential affine deformations of the image contents.

5.6 Feature detection

In computer vision and image processing the concept of feature detection refers to methods that aim at computing abstractions of image information and making local decisions at every image point whether there is an image feature of a given type at that point or not. The resulting features will be subsets of the image domain, often in the form of isolated points, continuous curves or connected regions.

5.6.1 Definition of a feature

There is no universal or exact definition of what constitutes a feature, and the exact definition often depends on the problem or the type of application. Given that, a feature is defined as an "interesting" part of an image, and features are used as a starting point for many computer vision algorithms. Since features are used

as the starting point and main primitives for subsequent algorithms, the overall algorithm will often only be as good as its feature detector. Consequently, the desirable property for a feature detector is repeatability: whether or not the same feature will be detected in two or more different images of the same scene.

Feature detection is a low-level image processing operation. That is, it is usually performed as the first operation on an image, and examines every pixel to see if there is a feature present at that pixel. If this is part of a larger algorithm, then the algorithm will typically only examine the image in the region of the features. As a built-in pre-requisite to feature detection, the input image is usually smoothed by a Gaussian kernel in a scale-space representation and one or several feature images are computed, often expressed in terms of local image derivatives operations.

Occasionally, when feature detection is computationally expensive and there are time constraints, a higher level algorithm may be used to guide the feature detection stage, so that only certain parts of the image are searched for features.

Many computer vision algorithms use feature detection as the initial step, so as a result, a very large number of feature detectors have been developed. These vary widely in the kinds of feature detected, the computational complexity and the repeatability.

5.6.2 Types of image features

- ***Edges:*** Edges are points where there is a boundary (or an edge) between two image regions. In general, an edge can be of almost arbitrary shape, and may include junctions. In practice, edges are usually defined as sets of points in the image which have a strong gradient magnitude. Furthermore, some common algorithms will then chain high gradient points together to form a more complete description of an edge. These algorithms usually place some constraints on the properties of an edge, such as shape, smoothness, and gradient value. Locally, edges have a one-dimensional structure.

- ***Corners / interest points:*** The terms corners and interest points are used somewhat interchangeably and refer to point-like features in an image, which have a local two dimensional structure. The name "Corner" arose since early

algorithms first performed edge detection, and then analysed the edges to find rapid changes in direction (corners). These algorithms were then developed so that explicit edge detection was no longer required, for instance by looking for high levels of curvature in the image gradient. It was then noticed that the so-called corners were also being detected on parts of the image which were not corners in the traditional sense (for instance a small bright spot on a dark background may be detected). These points are frequently known as interest points, but the term "corner" is used by tradition.

- ***Blobs / regions of interest points:*** Blobs provide a complementary description of image structures in terms of regions, as opposed to corners that are more point-like. Nevertheless, blob descriptors may often contain a preferred point (a local maximum of an operator response or a center of gravity) which means that many blob detectors may also be regarded as interest point operators. Blob detectors can detect areas in an image which are too smooth to be detected by a corner detector. Consider shrinking an image and then performing corner detection. The detector will respond to points which are sharp in the shrunk image, but may be smooth in the original image. It is at this point that the difference between a corner detector and a blob detector becomes somewhat vague. To a large extent, this distinction can be remedied by including an appropriate notion of scale. Nevertheless, due to their response properties to different types of image structures at different scales, the LoG and DoH blob detectors are also mentioned in the article on corner detection.

- ***Ridges:*** For elongated objects, the notion of ridges is a natural tool. A ridge descriptor computed from a grey-level image can be seen as a generalization of a medial axis. From a practical viewpoint, a ridge can be thought of as a one-dimensional curve that represents an axis of symmetry, and in addition has an attribute of local ridge width associated with each ridge point. Unfortunately, however, it is algorithmically harder to extract ridge features from general classes of grey-level images than edge-, corner- or blob features. Nevertheless, ridge descriptors are frequently used for road extraction in aerial images and for extracting blood vessels in medical images—see ridge detection.

5.6.3 Feature detectors

The operators included in this section are those whose purpose is to identify meaningful image features on the basis of distributions of pixel graylevels. The two categories of operators included here are:

- Edge Pixel Detectors, that assign a value to a pixel in proportion to the likelihood that the pixel is part of an image edge (i.e. a pixel that is on the boundary between two regions of different intensity values).
- Line Pixel Detectors, that assign a value to a pixel in proportion to the likelihood that the pixel is part of a image line (i.e. a dark narrow region bounded on both sides by lighter regions, or vice-versa).

Table 5.1: Common feature detectors and their classification:

Feature detector	Edge	Corner	Blob
Canny	X		
Sobel	X		
Kayyali	X		
Harris & Stephens / Plessey / Shi–Tomasi	X	X	
SUSAN	X	X	
Shi & Tomasi		X	
Level curve curvature		X	
FAST		X	X
Laplacian of Gaussian		X	X
Difference of Gaussians		X	X
Determinant of Hessian		X	X
MSER			X
PCBR			X
Grey-level blobs			X

5.6.4 Feature extraction

Once features have been detected, a local image patch around the feature can be extracted. This extraction may involve quite considerable amounts of image processing. The result is known as a feature descriptor or feature vector. Among the

approaches that are used to feature description, one can mention N-jets and local histograms (see scale-invariant feature transform for one example of a local histogram descriptor). In addition to such attribute information, the feature detection step by itself may also provide complementary attributes, such as the edge orientation and gradient magnitude in edge detection and the polarity and the strength of the blob in blob detection.

5.6.5 Good Features to Track

No feature-based vision system can work unless good features can be identified and tracked from frame to frame. Although tracking itself is by and large a solved problem, selecting features that can be tracked well and correspond to physical points in the world is still hard.

Harris corner detector performs well in many cases, but it misses out on a few things. Around six years after the original paper by Harris and Stephens, Shi-Tomasi came up with a better corner detector. You can read the original paper at [195]. They used a different scoring function to improve the overall quality. Using this method, we can find the N strongest corners in the given image. This is very useful when we don't want to use every single corner to extract information from the image.

5.7 Iterative Pyramidal Lucas–Kanade Optical Flow

The Lucas-Kanade approach is a local optimization problem that cannot perform properly if the object movements are too large. As the gradient information is obtained by neighboring pixels, the real object motion cannot extend beyond the considered region. Also, the local neighborhood taken into account for the least squares approach is finite and there are few chances to correctly determine large movements. Therefore, it is common to use a pyramidal implementation. The input images are resized to a lower resolution, first by filtering with a low pass filter and then subsampled by a factor of 2, technique called coarse-to-fine approach, as shown in Figure.5.16. The computation of the optical flow is started with the

lowest resolution images, at the highest pyramidal level. The result is passed then to the higher resolution level as an initial estimate. Running the algorithm on higher resolutions will cause higher accuracy for the flow field. Bouguet describes

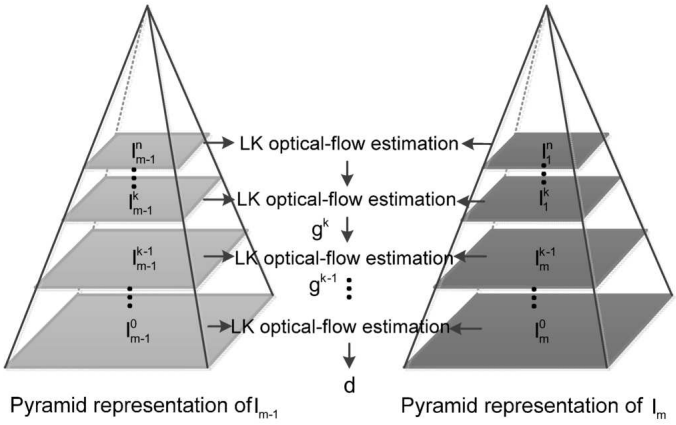


Figure 5.16: Coarse-to-fine optical flow estimation.

in [196] an iterative implementation of the Lucas-Kanade method using a Gaussian pyramid. The Iterative Lucas-Kanade algorithm requires an estimate of the velocity for every pixel using the classical algorithm. Then, by means of a warping technique, the estimated flow will be warped on the image and the process is repeated until convergence.

Chapter 6

Sensor Fusion

Sensor fusion is the process of using information from several different sensors to compute an estimate of the state of a dynamic system. The resulting estimate is in some sense better than it would be if the sensors were used individually. The term better can in this case mean more accurate, more reliable, more available and of higher safety integrity. Figure.**??** illustrates the aim of sensor fusion with some possible displacement sensing principles with typical properties. Advantages (+) of each are wished to be own by the fused signal (called estimation).

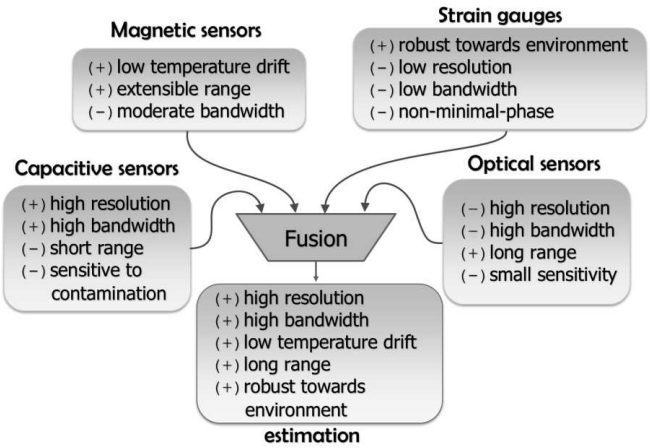


Figure 6.1: The aim of sensor fusion.

Sometimes a single sensor has a bandwidth smaller than required for high dynamic actuation. Besides it, sensors often suffer from drift and deliver signals superimposed with noise, however not in equal amounts in general. E.g. one sensor has high drift, by owning also high bandwidth, compared to another one with low drift and high resolution at a very limited bandwidth. The aim of sensor fusion is to take advantages of each sensor by combining (filtering) them individually in order to provide a position estimation according to the Performance criteria.

Furthermore, the resulting estimate may in some cases only be possible to obtain by using data from different types of sensors. Figure.**??** illustrates the basic concept of the sensor fusion framework. Many systems have traditionally been stand alone systems with one or several sensors transmitting information to only one single application. Using a sensor fusion approach it might be possible to remove one sensor and still perform the same tasks, or add new applications without the need to add new sensors.

Sensor fusion is required to reduce cost, system complexity and the number of components involved and to increase accuracy and confidence of sensing.

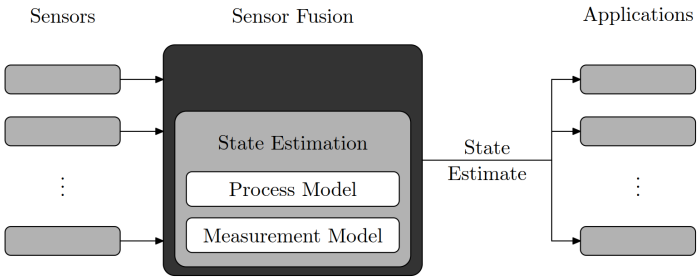


Figure 6.2: The main components of the sensor fusion framework are shown in the middle box. The framework receives measurements from several sensors, fuses them and produces one state estimate, which can be used by several applications.

6.1 Sensor fusion in robot manipulators

Open-chain manipulator robots are widely used in many industrial applications, such as painting and welding. Those applications require high performance in the

motion for increased quality in the manufactured products. The robotic system performance mainly relies on a good trajectory planning and efficient control. In the case of the trajectory planning, smooth movements are desired with constraints in the speed, acceleration and in some cases jerk [197]. Concerning to the motion controller, its performance depends on both of the control law and the feedback system. Most of these feedback systems are based on optical encoder [198]. Usually, they are attached to servomotors that control the angular position of each joint in the robot [199]. Those sensors provide accurate information about the servomotor position; however, they cannot detect mechanical imperfections and deformations that are common in open chain robots [200]. In some cases, depending of the method used to extrapolate velocity, the control could decrease its performance at low-velocity movements due to the operating principle of optical encoders [201]. Several proposed techniques can extract the velocity an acceleration information from the encoder but at expenses of adding phase delays that prevent its use in the controller [202].

Related with robotic systems, in [203] a set of controllers featuring speed feedback for serial robots are proposed. The proposal is tested in a 2-DOF robot with fast movements. Results show the importance of having joint-speed measurement available for the controller. Also, in [204] a disturbance observer is proposed for improving the performance of a n-link Robot. It is mentioned that acceleration signal is required to perform the disturbance observer, but such measurement is a hard task in many robotic applications. Other work [205] proposes a robust control based on multi-variable feedback and Lyapunov function. The proposal assumes that position, velocity and acceleration of the joints are available. In [200], a joint stiffness identification method for six-revolute serial robots is proposed. The mechanical problems common in serial robots are mentioned. They require translations and rotations measurements of the robot end effector. Also in [206] a filtering method is proposed for backlash estimation using accelerometer wrist torque sensor and position encoder information.

The mentioned works propose systems that attempts to improve the performance of motion controllers, also some works focuses on the estimation of mechanical imperfections in robots. However, most of the proposal assumes that velocity and

acceleration feedback is available, which is a difficult task in real applications. Concerning to this problem, researches propose sensor fusion techniques that attempts to improve the controller feedback. In [207], encoder and accelerometer information is fused through a combined technique of oversampling and average decimation filtering. Angular position, velocity, acceleration and jerk are estimated for single joint in a serial robot. Due to the decimation stage, the sampling rate of the proposal decreases at each derivative. An efficient technique for machine dynamics estimation based on optical encoder measurements is proposed in [208]. It consists of an adaptive high-order finite impulse response (FIR) differentiator that is defined as two impulse window filter (TIWF), which minimizes the quantization noise generated by successive derivatives of encoder information. The method features low computation; yet, there are induced delays that depend on the adaptive algorithm. In [209] the encoder and accelerometer fusion by means of particle filtering (PF) is proposed. Simulations are carried out for a 3 Degree-Of-Freedom (DOF) robot. They mention that PF outperforms Extended Kalman Filter (EKF) but at the expense of increasing the computational cost. This disadvantage could be prohibitive in industrial robots with higher DOF. A review on the accelerometer and gyroscope fusion for joint angle estimation is presented in [210]. In that work several fusion methods are compared and validated in a 3-DOF robot. In addition, the good cost-performance relation that MEMS-based (Microelectromechanical Systems-based) sensors provide is commented, as well as the needing of reliable methods for increasing the accuracy of the measurement. In [211], the high accuracy of optical encoders and the measurement continuity that accelerometers provide are fused by means of KF technique for joint angular estimation. The proposal is validated in a 6-DOF PUMA robot. However, the proposed methodology is not valid for the first joint where the accelerometer does not provide attitude information. In addition, the variables of angular velocity and acceleration are not calculated since the information from the utilized sensors is not enough to obtain a good estimation of such parameters.

The above mentioned techniques focus on providing accurate joint angle estimation in serial robots. Also, in some cases angular velocity is estimated. Nevertheless, only few works have been carried out for complete joint kinematics estimation, including angular acceleration, due to the fact that the accuracy in the measurement

system decreases with the successive derivatives [202]. Some works attempt to propose filtering techniques to overcome those problems; however, the sampling rate is compromised, inducing delays that avoid their use in robotic systems. For this reason, it would be desired to have a feedback system capable of providing the robot kinematic information of angular position, velocity and acceleration in each joint in serial robots. Moreover, through the fusion of adequate primary sensors it would be useful to provide the kinematics information without state delays.

6.2 Kalman Filter

Kalman filtering, also known as linear quadratic estimation (LQE), is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement alone, by using Bayesian inference and estimating a joint probability distribution over the variables for each timeframe. The filter is named after Rudolf E. Kálmán, one of the primary developers of its theory.

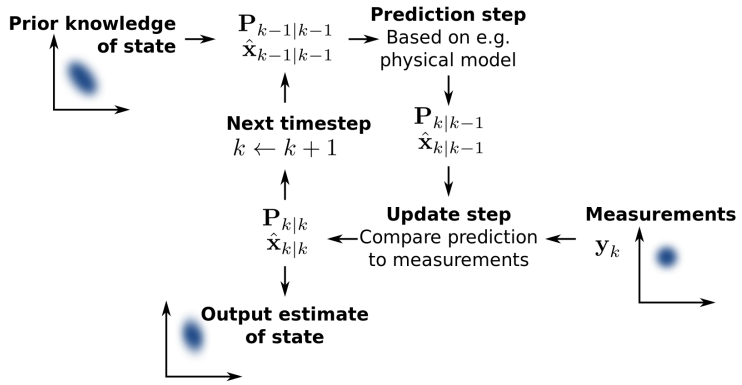


Figure 6.3: The diagram explains the basic steps of Kalman filtering: prediction and update. It also illustrates how the filter keeps track of not only the mean value of the state, but also the estimated variance.

The Kalman filter has numerous applications in technology. A common application is for data fusion of guidance, navigation variables for control of vehicles,

particularly aircraft and spacecraft [212]. Furthermore, the Kalman filter is a widely applied concept in time series analysis used in fields such as signal processing and econometrics. Kalman filters also are one of the main topics in the field of robotic motion planning and control, and they are sometimes included in trajectory optimization. The Kalman filter also works for modeling the central nervous system's control of movement. Due to the time delay between issuing motor commands and receiving sensory feedback, usage of the Kalman filter supports the realistic model for making estimates of the current state of the motor system and issuing updated commands [213].

The algorithm works in a two-step process, shown in figure.6.3. In the prediction step, the Kalman filter produces estimates of the current state variables, along with their uncertainties. Once the outcome of the next measurement (necessarily corrupted with some amount of error, including random noise) is observed, these estimates are updated using a weighted average, with more weight being given to estimates with higher certainty. The algorithm is recursive. It can run in real time, using only the present input measurements and the previously calculated state and its uncertainty matrix; no additional past information is required.

The Kalman filter does not make any assumption that the errors are Gaussian. However, the filter yields the exact conditional probability estimate in the special case that all errors are Gaussian-distributed. Extensions and generalizations to the method have also been developed, such as the extended Kalman filter and the unscented Kalman filter which work on nonlinear systems. The underlying model is a Bayesian model similar to a hidden Markov model except that the state space of the latent variables is continuous and all latent and observed variables have Gaussian distributions.

6.2.1 Overview of the calculation

The Kalman filter uses a system's dynamics model (e.g., physical laws of motion), known control inputs to that system, and multiple sequential measurements (such as from sensors) to form an estimate of the system's varying quantities (its state) that is better than the estimate obtained by using only one measurement alone. As such, it is a common sensor fusion and data fusion algorithm.

Noisy sensor data, approximations in the equations that describe the system evolution, and external factors that are not accounted for all place limits on how well it is possible to determine the system's state. The Kalman filter deals effectively with the uncertainty due to noisy sensor data and to some extent also with random external factors. The Kalman filter produces an estimate of the state of the system as an average of the system's predicted state and of the new measurement using a weighted average. The purpose of the weights is that values with better (i.e., smaller) estimated uncertainty are "trusted" more. The weights are calculated from the covariance, a measure of the estimated uncertainty of the prediction of the system's state. The result of the weighted average is a new state estimate that lies between the predicted and measured state, and has a better estimated uncertainty than either alone. This process is repeated at every time step, with the new estimate and its covariance informing the prediction used in the following iteration. This means that the Kalman filter works recursively and requires only the last "best guess", rather than the entire history, of a system's state to calculate a new state.

The relative certainty of the measurements and current state estimate is an important consideration, and it is common to discuss the response of the filter in terms of the Kalman filter's gain. The Kalman gain is the relative weight given to the measurements and current state estimate, and can be "tuned" to achieve particular performance. With a high gain, the filter places more weight on the most recent measurements, and thus follows them more responsively. With a low gain, the filter follows the model predictions more closely. At the extremes, a high gain close to one will result in a more jumpy estimated trajectory, while low gain close to zero will smooth out noise but decrease the responsiveness.

When performing the actual calculations for the filter (as discussed below), the state estimate and covariances are coded into matrices to handle the multiple dimensions involved in a single set of calculations. This allows for a representation of linear relationships between different state variables (such as position, velocity, and acceleration) in any of the transition models or covariances.

6.2.2 Underlying dynamical system model

The Kalman filters are based on linear dynamical systems discretized in the time domain. They are modelled on a Markov chain built on linear operators perturbed by errors that may include Gaussian noise. The state of the system is represented as a vector of real numbers. At each discrete time increment, a linear operator is applied to the state to generate the new state, with some noise mixed in, and optionally some information from the controls on the system if they are known. Then, another linear operator mixed with more noise generates the observed outputs from the true ("hidden") state. The Kalman filter may be regarded as analogous to the hidden Markov model, with the key difference that the hidden state variables take values in a continuous space (as opposed to a discrete state space as in the hidden Markov model). There is a strong duality between the equations of the Kalman Filter and those of the hidden Markov model.

In order to use the Kalman filter to estimate the internal state of a process given only a sequence of noisy observations, one must model the process in accordance with the framework of the Kalman filter. This means specifying the following matrices: F_k, the state-transition model; H_k, the observation model; Q_k, the covariance of the process noise; R_k, the covariance of the observation noise; and sometimes B_k, the control-input model, for each time-step, k, as described below.

The Kalman filter model assumes the true state at time k is evolved from the state at (k − 1) according to:

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{B}_k \mathbf{u}_k + \mathbf{w}_k \tag{6.1}$$

where:

- F_k is the state transition model which is applied to the previous state x_{k-1};
- B_k is the control-input model which is applied to the control vector u_k;
- w_k is the process noise which is assumed to be drawn from a zero mean multivariate normal distribution, $\mathcal{N}$, with covariance, Q_k: $\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k)$.

At time k an observation (or measurement) z_k of the true state x_k is made according to:

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \tag{6.2}$$

where:

- H_k is the observation model which maps the true state space into the observed space and
- v_k is the observation noise which is assumed to be zero mean Gaussian white noise with covariance R_k: $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$.

The initial state, and the noise vectors at each step $x_0, w_1, \ldots, w_k, v_1 \ldots v_k$ are all assumed to be mutually independent.

Many real dynamical systems do not exactly fit this model. In fact, unmodelled dynamics can seriously degrade the filter performance, even when it was supposed to work with unknown stochastic signals as inputs. The reason for this is that the effect of unmodelled dynamics depends on the input, and, therefore, can bring the estimation algorithm to instability (it diverges). On the other hand, independent white noise signals will not make the algorithm diverge. The problem of distinguishing between measurement noise and unmodelled dynamics is a difficult one and is treated in control theory under the framework of robust control.

The Kalman filter is a recursive estimator. This means that only the estimated state from the previous time step and the current measurement are needed to compute the estimate for the current state. In contrast to batch estimation techniques, no history of observations and/or estimates is required. In what follows, the notation $\hat{\mathbf{x}}_{n|m}$ represents the estimate of $\mathbf{x}$ at time n given observations up to and including at time $m \leqslant n$.

The state of the filter is represented by two variables:

- $\hat{\mathbf{x}}_{k|k}$, the a posteriori state estimate at time k given observations up to and including at time k;
- $\mathbf{P}_{k|k}$, the a posteriori error covariance matrix (a measure of the estimated accuracy of the state estimate).

The Kalman filter can be written as a single equation, however it is most often conceptualized as two distinct phases: "Predict" and "Update". The predict phase uses the state estimate from the previous timestep to produce an estimate of the state at the current timestep. This predicted state estimate is also known as the a priori state estimate because, although it is an estimate of the state at the current timestep, it does not include observation information from the current timestep. In the update phase, the current a priori prediction is combined with current observation information to refine the state estimate. This improved estimate is termed the a posteriori state estimate.

Typically, the two phases alternate, with the prediction advancing the state until the next scheduled observation, and the update incorporating the observation. However, this is not necessary; if an observation is unavailable for some reason, the update may be skipped and multiple prediction steps performed. Likewise, if multiple independent observations are available at the same time, multiple update steps may be performed (typically with different observation matrices H_k).

- ***Predict:***
  Predicted (a priori) state estimate:

$$\widehat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \widehat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k \mathbf{u}_k \tag{6.3}$$

  Predicted (a priori) estimate covariance:

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^{\mathrm{T}} + \mathbf{Q}_k \tag{6.4}$$

- ***Update:***
  Innovation or measurement pre-fit residual:

$$\widetilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \widehat{\mathbf{x}}_{k|k-1} \tag{6.5}$$

Innovation (or pre-fit residual) covariance:

$$\mathbf{S}_k = \mathbf{R}_k + \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^{\mathrm{T}} \tag{6.6}$$

Optimal Kalman gain:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^{\mathrm{T}} \mathbf{S}_k^{-1} \tag{6.7}$$

Updated (a posteriori) state estimate:

$$\widehat{\mathbf{x}}_{k|k} = \widehat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \widetilde{\mathbf{y}}_k \tag{6.8}$$

Updated (a posteriori) estimate covariance:

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} \tag{6.9}$$

Measurement post-fit residual:

$$\widetilde{\mathbf{y}}_{k|k} = \mathbf{z}_k - \mathbf{H}_k \widehat{\mathbf{x}}_{k|k} \tag{6.10}$$

The formula for the updated estimate covariance above is only valid for the optimal Kalman gain. Usage of other gain values requires a more complex formula found in the derivations section.

Invariants

If the model is accurate, and the values for $\hat{\mathbf{x}}_{0|0}$ and $\mathbf{P}_{0|0}$ accurately reflect the distribution of the initial state values, then the following invariants are preserved:

$$\begin{aligned} &\mathrm{E}[\mathbf{x}_k - \widehat{\mathbf{x}}_{k|k}] = \mathrm{E}[\mathbf{x}_k - \widehat{\mathbf{x}}_{k|k-1}] = 0 \\ &\mathrm{E}[\widetilde{\mathbf{y}}_k] = 0 \end{aligned} \tag{6.11}$$

where $\mathrm{E}[\xi]$ is the expected value of ξ . That is, all estimates have a mean error of

zero. Also:

$$\begin{aligned} \mathbf{P}_{k|k} &= \mathrm{cov}(\mathbf{x}_k - \widehat{\mathbf{x}}_{k|k}) \\ \mathbf{P}_{k|k-1} &= \mathrm{cov}(\mathbf{x}_k - \widehat{\mathbf{x}}_{k|k-1}) \\ \mathbf{S}_k &= \mathrm{cov}(\widetilde{\mathbf{y}}_k) \end{aligned} \tag{6.12}$$

so covariance matrices accurately reflect the covariance of estimates.

Estimation of the noise covariances Q_k ***and*** R_k***:***

Practical implementation of the Kalman Filter is often difficult due to the difficulty of getting a good estimate of the noise covariance matrices Q_k and R_k. Extensive research has been done in this field to estimate these covariances from data. One practical approach to do this is the autocovariance least-squares (ALS) technique that uses the time-lagged autocovariances of routine operating data to estimate the covariances.The GNU Octave and Matlab code used to calculate the noise covariance matrices using the ALS technique is available online under the GNU General Public License.

Optimality and performance:

It follows from theory that the Kalman filter is the optimal linear filter in cases where a) the model perfectly matches the real system, b) the entering noise is white (uncorrelated) and c) the covariances of the noise are exactly known. Several methods for the noise covariance estimation have been proposed during past decades, including ALS, mentioned in the section above. After the covariances are estimated, it is useful to evaluate the performance of the filter; i.e., whether it is possible to improve the state estimation quality. If the Kalman filter works optimally, the innovation sequence (the output prediction error) is a white noise, therefore the whiteness property of the innovations measures filter performance. Several different methods can be used for this purpose. If the noise terms are non-Gaussian distributed, methods for assessing performance of the filter estimate, which use probability inequalities or large-sample theory, are known in the literature.

6.3 Sliding Mode Observer

Sliding mode techniques were perhaps originally best known for their potential as a robust control method, and evolved from pioneering work in the 1960s in the former

Soviet Union. In control systems, sliding mode control, or SMC, is a nonlinear control method that alters the dynamics of a nonlinear system by application of a discontinuous control signal (or more rigorously, a set-valued control signal) that forces the system to "slide" along a cross-section of the system's normal behavior. The state-feedback control law is not a continuous function of time. Instead, it can switch from one continuous structure to another based on the current position in the state space. Hence, sliding mode control is a variable structure control method. The multiple control structures are designed so that trajectories always move toward an adjacent region with a different control structure, and so the ultimate trajectory will not exist entirely within one control structure. Instead, it will slide along the boundaries of the control structures. The motion of the system as it slides along these boundaries is called a sliding mode and the geometrical locus consisting of the boundaries is called the sliding (hyper)surface. In the context of modern control theory, any variable structure system, like a system under SMC, may be viewed as a special case of a hybrid dynamical system as the system both flows through a continuous state space but also moves through different discrete control modes.

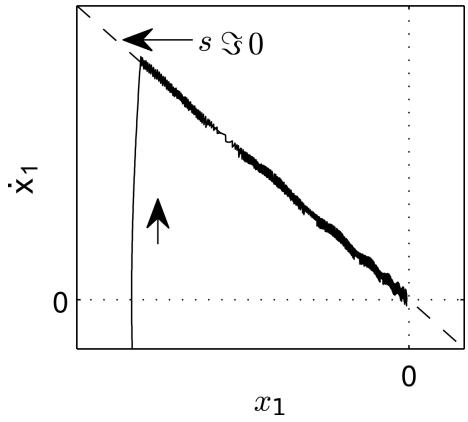


Figure 6.4: Phase plane trajectory of a system being stabilized by a sliding mode controller. After the initial reaching phase, the system states "slides" along the line $s = 0$. The particular $s = 0$ surface is chosen because it has desirable reduced-order dynamics when constrained to it. In this case, the $s = x_1 + \dot{x}_1 = 0$ surface corresponds to the first-order LTI system $\dot{x}_1 = -x_1$, which has an exponentially stable origin.

6.3.1 Introduction

$s = 0$, and the sliding mode along the surface commences after the finite time when system trajectories have reached the surface. In the theoretical description of sliding modes, the system stays confined to the sliding surface and need only be viewed as sliding along the surface. However, real implementations of sliding mode control approximate this theoretical behavior with a high-frequency and generally non-deterministic switching control signal that causes the system to "chatter" in a tight neighborhood of the sliding surface. In fact, although the system is nonlinear in general, the idealized (i.e., non-chattering) behavior of the system in Figure.6.4 when confined to the $s = 0$ surface is an LTI system with an exponentially stable origin.

Intuitively, sliding mode control uses practically infinite gain to force the trajectories of a dynamic system to slide along the restricted sliding mode subspace. Trajectories from this reduced-order sliding mode have desirable properties (e.g., the system naturally slides along it until it comes to rest at a desired equilibrium). The main strength of sliding mode control is its robustness. Because the control can be as simple as a switching between two states (e.g., "on"/"off" or "forward"/"reverse"), it need not be precise and will not be sensitive to parameter variations that enter into the control channel. Additionally, because the control law is not a continuous function, the sliding mode can be reached in finite time (i.e., better than asymptotic behavior). Under certain common conditions, optimality requires the use of bang–bang control; hence, sliding mode control describes the optimal controller for a broad set of dynamic systems.

One application of sliding mode controller is the control of electric drives operated by switching power converters. Because of the discontinuous operating mode of those converters, a discontinuous sliding mode controller is a natural implementation choice over continuous controllers that may need to be applied by means of pulse-width modulation or a similar technique of applying a continuous signal to an output that can only take discrete states. Sliding mode control has many applications in robotics. In particular, this control algorithm has been used for tracking control of unmanned surface vessels in simulated rough seas with high degree of success.

Sliding mode control must be applied with more care than other forms of nonlinear control that have more moderate control action. In particular, because actuators have delays and other imperfections, the hard sliding-mode-control action can lead to chatter, energy loss, plant damage, and excitation of unmodeled dynamics. Continuous control design methods are not as susceptible to these problems and can be made to mimic sliding-mode controllers.

6.3.2 Control scheme

Consider a nonlinear dynamical system described by:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}, t) + B(\mathbf{x}, t)\mathbf{u}(t) \tag{6.13}$$

where:

$$\mathbf{x}(t) \triangleq \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_{n-1}(t) \\ x_n(t) \end{bmatrix} \in \mathbb{R}^n \tag{6.14}$$

is an n-dimensional state vector and:

$$\mathbf{u}(t) \triangleq \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_{m-1}(t) \\ u_m(t) \end{bmatrix} \in \mathbb{R}^m \tag{6.15}$$

is an m-dimensional input vector that will be used for state feedback. The functions $f : \mathbb{R}^n \times \mathbb{R} \mapsto \mathbb{R}^n$ and $B : \mathbb{R}^n \times \mathbb{R} \mapsto \mathbb{R}^{n \times m}$ are assumed to be continuous and sufficiently smooth so that the Picard–Lindelöf theorem can be used to guarantee that solution $\mathbf{x}(t)$ to Equation.6.13 exists and is unique.

A common task is to design a state-feedback control law $\mathbf{u}(\mathbf{x}(t))$ (i.e., a mapping from current state $\mathbf{x}(t)$ at time t to the input $\mathbf{u}$ to stabilize the dynamical system in Equation.6.13 around the origin $\mathbf{x} = [0, 0, \ldots, 0]^{\mathrm{T}}$. That is, under the

control law, whenever the system is started away from the origin, it will return to it. For example, the componentx_1 of the state vector $\mathbf{x}$ may represent the difference some output is away from a known signal (e.g., a desirable sinusoidal signal); if the control $\mathbf{u}$ can ensure that x_1 quickly returns to $x_1 = 0$, then the output will track the desired sinusoid. In sliding-mode control, the designer knows that the system behaves desirably (e.g., it has a stable equilibrium) provided that it is constrained to a subspace of its configuration space. Sliding mode control forces the system trajectories into this subspace and then holds them there so that they slide along it. This reduced-order subspace is referred to as a sliding (hyper)surface, and when closed-loop feedback forces trajectories to slide along it, it is referred to as a sliding mode of the closed-loop system. Trajectories along this subspace can be likened to trajectories along eigenvectors (i.e., modes) of LTI systems; however, the sliding mode is enforced by creasing the vector field with high-gain feedback. Like a marble rolling along a crack, trajectories are confined to the sliding mode.

The sliding-mode control scheme involves:

- Selection of a hypersurface or a manifold (i.e., the sliding surface) such that the system trajectory exhibits desirable behavior when confined to this manifold.
- Finding feedback gains so that the system trajectory intersects and stays on the manifold.

Because sliding mode control laws are not continuous, it has the ability to drive trajectories to the sliding mode in finite time (i.e., stability of the sliding surface is better than asymptotic). However, once the trajectories reach the sliding surface, the system takes on the character of the sliding mode (e.g., the origin $\mathbf{x} = \mathbf{0}$ may only have asymptotic stability on this surface).

The sliding-mode designer picks a switching function $s : \mathbb{R}^n \mapsto \mathbb{R}^m$ that represents a kind of "distance" that the states $\mathbf{x}$ are away from a sliding surface.

- A state $\mathbf{x}$ that is outside of this sliding surface has $s(\mathbf{x}) \neq 0$.
- A state that is on this sliding surface has $s(\mathbf{x}) = 0$.

The sliding-mode-control law switches from one state to another based on the sign of this distance. So the sliding-mode control acts like a stiff pressure always pushing in the direction of the sliding mode where $s(\mathbf{x}) = 0$. Desirable $\mathbf{x}(t)$ trajectories will approach the sliding surface, and because the control law is not continuous (i.e., it switches from one state to another as trajectories move across this surface), the surface is reached in finite time. Once a trajectory reaches the surface, it will slide along it and may, for example, move toward the $\mathbf{x} = \mathbf{0}$ origin. So the switching function is like a topographic map with a contour of constant height along which trajectories are forced to move.

The sliding (hyper)surface is of dimension $n \times m$ where n is the number of states in $\mathbf{x}$ and m is the number of input signals (i.e., control signals) in $\mathbf{u}$. For each control index $1 \leq k \leq m$, there is an $n \times 1$ sliding surface given by:

$$\{\mathbf{x} \in \mathbb{R}^n : \sigma_k(\mathbf{x}) = 0\} \tag{6.16}$$

The vital part of SMC design is to choose a control law so that the sliding mode (i.e., this surface given by $s(\mathbf{x}) = \mathbf{0}$) exists and is reachable along system trajectories. The principle of sliding mode control is to forcibly constrain the system, by suitable control strategy, to stay on the sliding surface on which the system will exhibit desirable features. When the system is constrained by the sliding control to stay on the sliding surface, the system dynamics are governed by reduced-order system obtained from Equation.6.16.

To force the system states $\mathbf{x}$ to satisfy $s(\mathbf{x}) = \mathbf{0}$, one must:

- Ensure that the system is capable of reaching $s(\mathbf{x}) = \mathbf{0}$ from any initial condition.
- Having reached $s(\mathbf{x}) = \mathbf{0}$, the control action is capable of maintaining the system at $s(\mathbf{x}) = \mathbf{0}$.

6.3.3 Existence of closed-loop solutions

Note that because the control law is not continuous, it is certainly not locally Lipschitz continuous, and so existence and uniqueness of solutions to the closed-loop system is not guaranteed by the Picard–Lindelöf theorem. Thus the solutions are

to be understood in the Filippov sense. Roughly speaking, the resulting closed-loop system moving along $s(\mathbf{x}) = \mathbf{0}$ is approximated by the smooth dynamics $\dot{s}(\mathbf{x}) = \mathbf{0}$; however, this smooth behavior may not be truly realizable. Similarly, high-speed pulse-width modulation or delta-sigma modulation produces outputs that only assume two states, but the effective output swings through a continuous range of motion. These complications can be avoided by using a different nonlinear control design method that produces a continuous controller. In some cases, sliding-mode control designs can be approximated by other continuous control designs.

6.3.4 Observer Mode

Sliding mode control can be used in the design of state observers. These non-linear high-gain observers have the ability to bring coordinates of the estimator error dynamics to zero in finite time. Additionally, switched-mode observers have attractive measurement noise resilience that is similar to a Kalman filter. For simplicity, the example here uses a traditional sliding mode modification of a Luenberger observer for an LTI system. In these sliding mode observers, the order of the observer dynamics are reduced by one when the system enters the sliding mode. In this particular example, the estimator error for a single estimated state is brought to zero in finite time, and after that time the other estimator errors decay exponentially to zero. However, as first described by Drakunov, a sliding mode observer for non-linear systems can be built that brings the estimation error for all estimated states to zero in a finite (and arbitrarily small) time.

Here, consider the LTI system:

$$\begin{cases} \dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \\ y = \begin{bmatrix} 1 & 0 & 0 & \cdots \end{bmatrix} \mathbf{x} = x_1 \end{cases} \tag{6.17}$$

where state vector $\mathbf{x} \triangleq (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$, $\mathbf{u} \triangleq (u_1, u_2, \ldots, u_r) \in \mathbb{R}^r$ is a vector of inputs, and output y is a scalar equal to the first state of the $\mathbf{x}$ state vector. Let:

$$A \triangleq \begin{bmatrix} a_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \tag{6.18}$$

where:

- a_{11} is a scalar representing the influence of the first state x_1 on itself,
- $A_{21} \in \mathbb{R}^{(n-1)}$ is a column vector representing the influence of the other states on the first state,
- $A_{22} \in \mathbb{R}^{(n-1)\times(n-1)}$ is a matrix representing the influence of the other states on themselves, and
- $A_{12} \in \mathbb{R}^{1\times(n-1)}$ is a row vector corresponding to the influence of the first state on the other states.

The goal is to design a high-gain state observer that estimates the state vector $\mathbf{x}$ using only information from the measurement $y = x_1$. Hence, let the vector $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n) \in \mathbb{R}^n$ be the estimates of the n states. The observer takes the form:

$$\dot{\widehat{\mathbf{x}}} = A\widehat{\mathbf{x}} + B\mathbf{u} + Lv(\hat{x}_1 - x_1) \tag{6.19}$$

where $v : \mathbb{R} \mapsto \mathbb{R}$ is a nonlinear function of the error between estimated state $\hat{x}_1$ and the output $y = x_1$, and $L \in \mathbb{R}^n$ is an observer gain vector that serves a similar purpose as in the typical linear Luenberger observer. Likewise, let:

$$L = \begin{bmatrix} -1 \\ L_2 \end{bmatrix} \tag{6.20}$$

where $L_2 \in \mathbb{R}^{(n-1)}$ is a column vector. Additionally, let $\mathbf{e} = (e_1, e_2, \ldots, e_n) \in \mathbb{R}^n$ be the state estimator error. That is, $\mathbf{e} = \hat{\mathbf{x}} - \mathbf{x}$. The error dynamics are then:

$$\begin{cases} \dot{\mathbf{e}} = \dot{\widehat{\mathbf{x}}} - \dot{\mathbf{x}} \\ = A\widehat{\mathbf{x}} + B\mathbf{u} + Lv(\hat{x}_1 - x_1) - A\mathbf{x} - B\mathbf{u} \\ = A(\widehat{\mathbf{x}} - \mathbf{x}) + Lv(\hat{x}_1 - x_1) \\ = A\mathbf{e} + Lv(e_1) \end{cases} \tag{6.21}$$

where $e_1 = \hat{x}_1 - x_1$ is the estimator error for the first state estimate. The nonlinear control law v can be designed to enforce the sliding manifold:

$$0 = \hat{x}_1 - x_1 \tag{6.22}$$

so that estimate $\hat{x}_1$ tracks the real state x_1 after some finite time (i.e., $\hat{x}_1 = x_1$). Hence, the sliding mode control switching function:

$$s(\hat{x}_1, \hat{x}) \triangleq e_1 = \hat{x}_1 - x_1. \tag{6.23}$$

To attain the sliding manifold, $\dot{s}$ and s must always have opposite signs (i.e., $s\dot{s} < 0$ for essentially all $\mathbf{x}$). However,

$$\dot{s} = \dot{e}_1 = a_{11}e_1 + A_{12}\mathbf{e}_2 - v(e_1) = a_{11}e_1 + A_{12}\mathbf{e}_2 - v(s) \tag{6.24}$$

where $\mathbf{e}_2 \triangleq (e_2, e_3, \ldots, e_n) \in \mathbb{R}^{(n-1)}$ is the collection of the estimator errors for all of the unmeasured states. To ensure that $s\dot{s} < 0$, let:

$$v(s) = M\,\mathrm{sgn}(s) \tag{6.25}$$

where:

$$M > \max\{|a_{11}e_1 + A_{12}\mathbf{e}_2|\} \tag{6.26}$$

That is, positive constant M must be greater than a scaled version of the maximum possible estimator errors for the system (i.e., the initial errors, which are assumed to be bounded so that M can be picked large enough; al). If M is sufficiently large, it can be assumed that the system achieves $e_1 = 0$ (i.e., $\hat{x}_1 = x_1$). Because e_1 is constant (i.e., 0) along this manifold, $\dot{e}_1 = 0$ as well. Hence, the discontinuous control $v(s)$ may be replaced with the equivalent continuous control v_{eq} where:

$$0 = \dot{s} = a_{11}\overbrace{e_1}^{=0} + A_{12}\mathbf{e}_2 - \overbrace{v_{\mathrm{eq}}}^{v(s)} = A_{12}\mathbf{e}_2 - v_{\mathrm{eq}} \tag{6.27}$$

So:

$$\overbrace{v_{\mathrm{eq}}}^{\text{scalar}} = \overbrace{A_{12}}^{1\times(n-1)\text{ vector}} \overbrace{\mathbf{e}_2}^{(n-1)\times 1\text{ vector}}. \tag{6.28}$$

This equivalent control v_{eq} represents the contribution from the other $(n-1)$ states to the trajectory of the output state x_1. In particular, the row A_{12} acts like an

output vector for the error subsystem:

$$\overbrace{\begin{bmatrix} \dot{e}_2 \\ \dot{e}_3 \\ \vdots \\ \dot{e}_n \end{bmatrix}}^{\dot{\mathbf{e}}_2} = A_2 \overbrace{\begin{bmatrix} e_2 \\ e_3 \\ \vdots \\ e_n \end{bmatrix}}^{\mathbf{e}_2} + L_2 v(e_1) = A_2\mathbf{e}_2 + L_2 v_{\text{eq}} = A_2\mathbf{e}_2 + L_2 A_{12}\mathbf{e}_2 = (A_2 + L_2 A_{12})\mathbf{e}_2. \tag{6.29}$$

So, to ensure the estimator error $\mathbf{e}_2$ for the unmeasured states converges to zero, the $(n-1)\times 1$ vector L_2 must be chosen so that the$(n-1)\times(n-1)$ matrix $(A_2+L_2A_{12})$is Hurwitz (i.e., the real part of each of its eigenvalues must be negative). Hence, provided that it is observable, this $\mathbf{e}_2$ system can be stabilized in exactly the same way as a typical linear state observer when A_{12} is viewed as the output matrix (i.e., " C"). That is, the v_{eq} equivalent control provides measurement information about the unmeasured states that can continually move their estimates asymptotically closer to them. Meanwhile, the discontinuous control $v = M\,\text{sgn}(\hat{x}_1 - x)$ forces the estimate of the measured state to have zero error in finite time. Additionally, white zero-mean symmetric measurement noise (e.g., Gaussian noise) only affects the switching frequency of the control v, and hence the noise will have little effect on the equivalent sliding mode control v_{eq}. Hence, the sliding mode observer has Kalman filter–like features [214].

The final version of the observer is thus:

$$\begin{cases} \dot{\widehat{\mathbf{x}}} = A\widehat{\mathbf{x}} + B\mathbf{u} + LM\,\text{sgn}(\hat{x}_1 - x_1) \\ = A\widehat{\mathbf{x}} + B\mathbf{u} + \begin{bmatrix} -1 \\ L_2 \end{bmatrix} M\,\text{sgn}(\hat{x}_1 - x_1) \\ = A\widehat{\mathbf{x}} + B\mathbf{u} + \begin{bmatrix} -M \\ L_2 M \end{bmatrix} \text{sgn}(\hat{x}_1 - x_1) \\ = A\widehat{\mathbf{x}} + \begin{bmatrix} B & \begin{bmatrix} -M \\ L_2 M \end{bmatrix} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \text{sgn}(\hat{x}_1 - x_1) \end{bmatrix} \\ = A_{\text{obs}}\widehat{\mathbf{x}} + B_{\text{obs}}\mathbf{u}_{\text{obs}} \end{cases} \tag{6.30}$$

where:

$$\begin{aligned} A_{\text{obs}} &\triangleq A \\ B_{\text{obs}} &\triangleq \begin{bmatrix} B & \begin{bmatrix} -M \\ L_2 M \end{bmatrix} \end{bmatrix} \\ u_{\text{obs}} &\triangleq \begin{bmatrix} \mathbf{u} \\ \text{sgn}(\hat{x}_1 - x_1) \end{bmatrix} \end{aligned} \tag{6.31}$$

That is, by augmenting the control vector $\mathbf{u}$ with the switching function $\text{sgn}(\hat{x}_1 - x_1)$, the sliding mode observer can be implemented as an LTI system. That is, the discontinuous signal $\text{sgn}(\hat{x}_1 - x_1)$ is viewed as a control input to the 2-input LTI system. For simplicity, this example assumes that the sliding mode observer has access to a measurement of a single state (i.e., output $y = x_1$). However, a similar procedure can be used to design a sliding mode observer for a vector of weighted combinations of states (i.e., when output $\mathbf{y} = C\mathbf{x}$ uses a generic matrix C). In each case, the sliding mode will be the manifold where the estimated output $\hat{\mathbf{y}}$ follows the measured output $\mathbf{y}$ with zero error (i.e., the manifold where $s(\mathbf{x}) \triangleq \hat{\mathbf{y}} - \mathbf{y} = \mathbf{0}$).

Part III

Contributions

Chapter 7

Advanced Robotic Manipulator Simulator "ADROMS"

Robot manipulators are the main devices of technology and industrial advancements in few recent decades [215, 216]. They are hands of man in technology and so need to be as precise, quick and trustworthy as we need them. Manipulators are being taught in academic extensively but not so many practices because of their prices. New ideas come in mind of students when reading books and articles but as lack of experimental laboratories they could not work on them.

One special aspect of manipulators is designing control strategies and applying them on real robots. This aspect needs more special treatment as it is the key point to accuracy and repeatability of each robot. Many softwares and toolboxes are designed to help students of mechatronic to learn basics of robot manipulators. But none of them are focused in a special aspect like control engineering designing and many of them are developed just for some special cases with limited DOF manipulators.

In control designing process, knowing of the fully parametric structure of the dynamics of any kind of robot manipulator is needed to study and design a successful strategy. As these requirements, the Advanced Robotic Manipulator Simulator "ADROMS" has been developed in University of Bergamo for studying on advanced robotics. ADROMS is a fully parametric- symbolic universal serial robot manipulator simulator which is written in MATLAB for control developments. Being the

most automatic-ware and simultaneously flexible in practice were of the development goals. In this paper the toolbox and its components will be described.

7.1 SCARA Simulation

This chapter deals with simulations of the dynamic model in open loop and closed loop. Note that the goal is to perform simulations to SCARA manipulator in order to prove the validity of the model and control system.

7.1.1 Brief History

The SCARA Fig.(7.1) acronym stands for Selective Compliance Assembly Robot Arm or Selective Compliance Articulated Robot Arm. In 1981, Sankyo Seiki, Pentel and NEC presented a completely new concept for assembly robots. The robot was developed under the guidance of Hiroshi Makino, a professor at the University of Yamanashi. The robot was called Selective Compliance Assembly Robot Arm, SCARA. Its arm was rigid in the Z-axis and pliable in the XY-axes, which allowed it to adapt to holes in the XY-axes.By virtue of the SCARA's parallel-axis joint layout, the arm is slightly compliant in the X-Y direction but rigid in the 'Z' direction, hence the term: Selective Compliant. This is advantageous for many types of assembly operations, i.e., inserting a round pin in a round hole without binding. The second attribute of the SCARA is the jointed two-link arm layout similar to our human arms, hence the often-used term, Articulated. This feature allows the arm to extend into confined areas and then retract or "fold up" out of the way. This is advantageous for transferring parts from one cell to another or for loading/ unloading process stations that are enclosed. SCARAs are generally faster and cleaner than comparable Cartesian robot systems. Their single pedestal mount requires a small footprint and provides an easy, unhindered form of mounting. On the other hand, SCARAs can be more expensive than comparable Cartesian systems and the controlling software requires inverse kinematics for linear interpolated moves. This software typically comes with the SCARA though and is usually transparent to the end-user.In addition to the 30-50% higher throughput with its motion blending feature, SCARA also provides reduced tool footprint for many applications.

Figure 7.1: An EPSON Scara LS3: standard version.

7.1.2 SCARA construction Step-by-Step

Conventional SCARA manipulator is made up of one pillar, a planar manipulator and a prismatic end-effector Fig.(7.2).

7.1.2.1 Table Structure

By considering ADROMS principles, the Denavit-Hartenberg structure of this manipulator can be constructed as Fig.(7.3). In this structure at first the full structure would be found by symbolic ADROMS and then the 2 variables of joint 4, L_4 and b_4 will be zeroed. By considering those principles we write the Denavit-Hartenberg table of this robot as:
DHTable=[q(1),h,0,L(1);q(2),0,0,L(2);0,-L(3)+q(3),0,0;q(4),0,0,L(4)];

This DH table have 4 rows for each variable q_i. In the first row we have full description of the linkage with mass activated in the radius part of pillar. This in fact is just an elevation for the next coordinates. Second row is the planar manipulator linkage and the forth row is down-slided planar linkage. In the third row a pillar with maximum L_3 length has been described which a prismatic variable stands on it and varies from 0 to L_3.

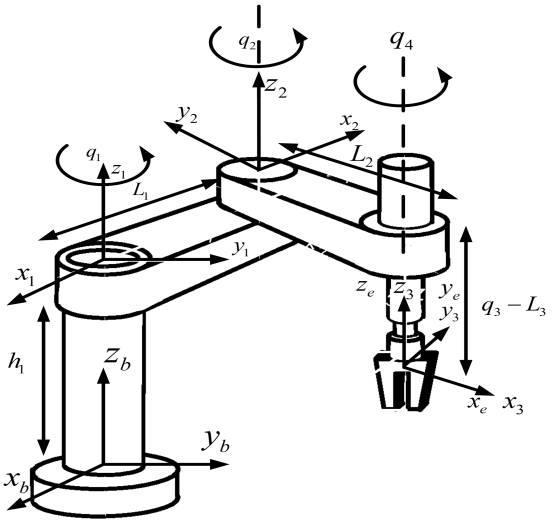


Figure 7.2: Conventional SCARA manipulator with parameters.

7.1.2.2 Generated Symbolic Position and Kinematic Matrices

In script e006_001 _SCARA_Symbolic.m the kinematic and dynamics of the SCARA manipulator has been derived.

Position, Rotation and Kinematic matrices are:

$$Pos = \begin{matrix} \mathrm{L2} * \cos(\mathrm{q1} + \mathrm{q2}) + \mathrm{L1} * \cos(\mathrm{q1}) \\ \mathrm{L2} * \sin(\mathrm{q1} + \mathrm{q2}) + \mathrm{L1} * \sin(\mathrm{q1}) \\ \mathrm{h} - \mathrm{L3} + \mathrm{q3} \end{matrix}$$

$$Rot = \begin{matrix} [\cos(\mathrm{q1} + \mathrm{q2} + \mathrm{q4}), -\sin(\mathrm{q1} + \mathrm{q2} + \mathrm{q4}), 0] \\ [\sin(\mathrm{q1} + \mathrm{q2} + \mathrm{q4}), \cos(\mathrm{q1} + \mathrm{q2} + \mathrm{q4}), 0] \\ [0, 0, 1] \end{matrix}$$

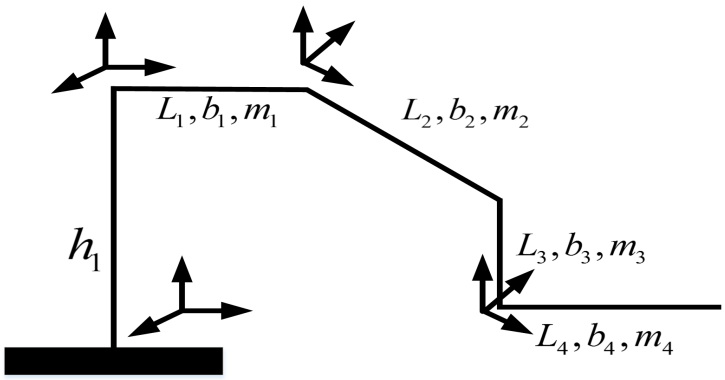


Figure 7.3: Structure of a SCARA in ADROMS.

$$Jac = \begin{matrix} [-\text{L2} * \sin(\text{q1} + \text{q2}) - \text{L1} * \sin(\text{q1}), -\text{L2} * \sin(\text{q1} + \text{q2}), 0, 0] \\ [\text{L2} * \cos(\text{q1} + \text{q2}) + \text{L1} * \cos(\text{q1}), \text{L2} * \cos(\text{q1} + \text{q2}), 0, 0] \\ [0, 0, 1, 0] \\ [0, 0, 0, 0] \\ [0, 0, 0, 0] \\ [1, 1, 0, 1] \end{matrix}$$

7.1.3 Path Planning for SCARA manipulator

For path planning of a robot, the workspace of it must be known. In the ADROMS there is a function to plot 3D work-space of any supported manipulator in good time. In Fig.(7.4) the work-space of the SCARA robot has been drawn.

7.1.3.1 Dynamic Nonlinear Optimization for smooth path planning

Consider forward kinematic for SCARA manipulator. Then we can construct a nonlinear optimal objective function as defined in chapter 2. We assume that each joint just can move from $-\pi/3$ to $\pi - \pi/6$ radian and the joint 3 can move from 0 to L_3.

The position that has been considered for construction is a square in XY and sin

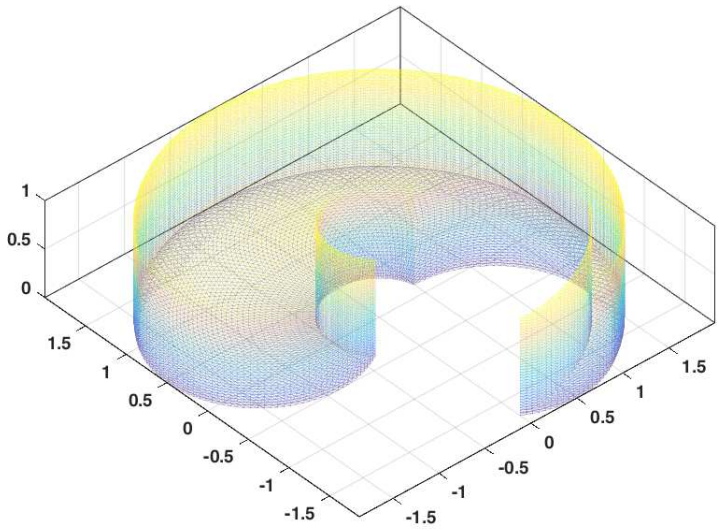


Figure 7.4: Work-space of a SCARA in ADROMS.

wave in Z dimention which take 20[sec] to complete:

$$\begin{aligned} x &= d(|\cos(tp)|\cos(tp) - |\sin(tp)|\sin(tp)) \\ y &= p(|\cos(tp)|\cos(tp) + |\sin(tp)|\sin(tp)) + 1.5 \\ z &= 0.5\sin(2\pi 0.2tp) + 0.5 \end{aligned}$$

Beside this path, two more paths has been considered. The first One is the closest path from the initial robot standing point to starting point of the work-path in 3[sec]. The second is the initial resting at starting point of the work-path which takes 1[sec]. These paths and the found smooth constraint path in joint-space has been shown in Fig.(7.5)

In Fig.(7.6) the found joint-space path has been shown. As illustrated, it is smooth and constrained. As joint 3 does not have any influence in positioning of the end-effector, it is found to be constant.

This found path has maximum $70\mu meter$ from reference path. All the errors from reference path points has been shown in Fig.(7.7). This process is good enough for

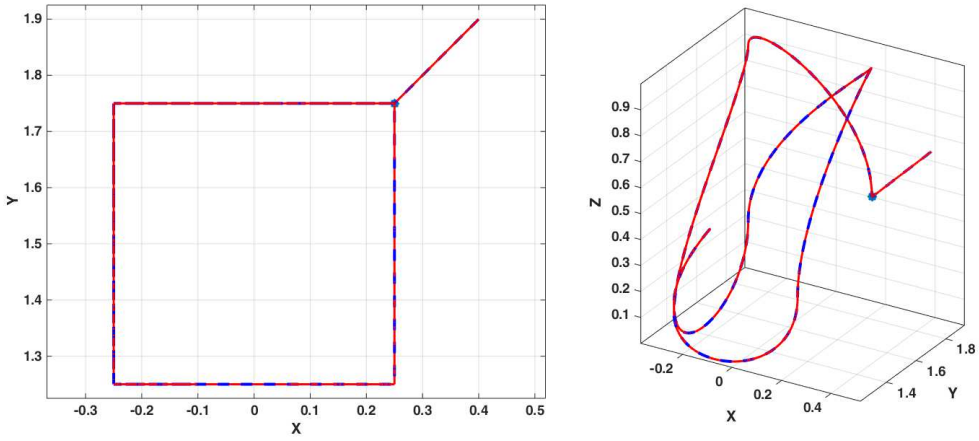


Figure 7.5: Work-path: left)from XY view. right) 3D view. Relax point is shown with a star.

fixed light to medium robotic CNC machine path planning so the inside servomotors of each joint can follow the joint space variables as planed.

7.1.4 Sliding Mode Control with additional Integral correction

Ongoing industrial processes needs every day, more investment on large scale and simultaneously light weighted tools to produce much large scale products. In this section a control of a heavy duty SCARA will be studied.

7.1.4.1 Sliding Mode Control

In control system, sliding mode control, or SMC, is a nonlinear control method that alters the dynamics of a nonlinear system by application of a discontinuous control signal that forces the system to "slide" along a cross-section of the system's normal behavior. The state-feedback control law is not a continuous function of time. Instead, it can switch from one continuous structure to another based on the current position in the state space. Hence, sliding mode control is a variable structure control method. The multiple control structures are designed so that trajectories

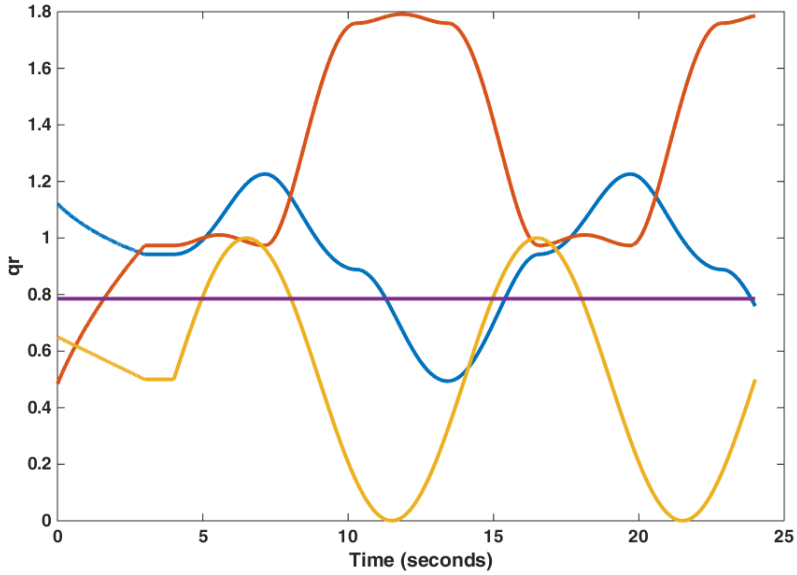


Figure 7.6: Joint-space path found by optimizing optimal path equation.

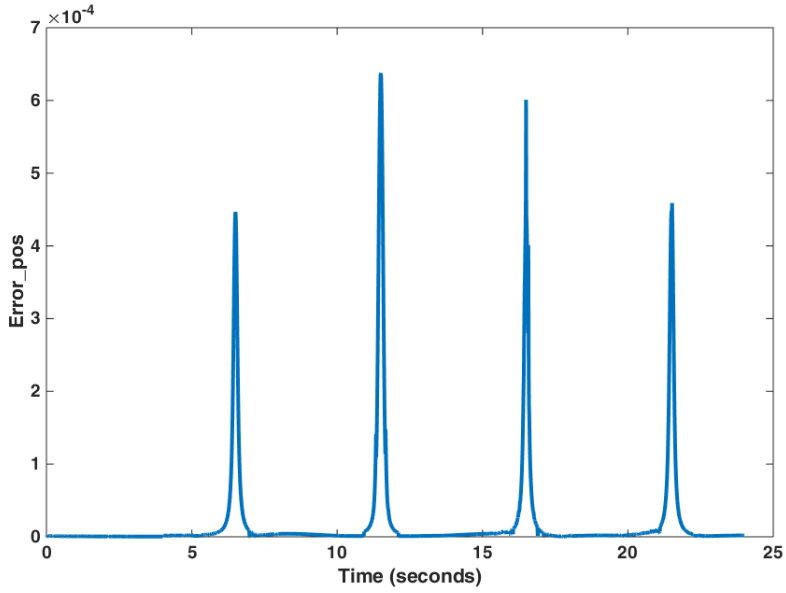


Figure 7.7: Found path error with respect to reference path.

always move toward an adjacent region with a different control structure, and so the ultimate trajectory will not exist entirely within one control structure. Instead, it will slide along the boundaries of the control structures. The motion of the system as it slides along these boundaries is called a sliding mode and the geometrical locus consisting of the boundaries is called the sliding (hyper)surface. In the context of modern control theory, any variable structure system, like a system under SMC, may be viewed as a special case of a hybrid dynamical system as the system both flows through a continuous state space but also moves through different discrete control modes.

7.1.4.2 Dynamic Model of Robot Manipulator

Full dynamic of a Robot Manipulator can be written as:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau_{\text{internal}} - \tau_{external} - \tau_{frictions}$$

In the ADROMS toolbox, the function [M,C,P,Jac,Pos,Rotj,Jw]= symDynamic (m,q, dq,L,b,Ma,Jm,DHTable,g) computes dynamic model of all the types of serial manipulators based on Denavit-Hartenberg table symbolically which is the most needed in control strategy designs. In e006_001_SCARA _Symbolic.m full dynamic of SACARA manipulator has been derived.

The M matrix is:

$M_{11} = J1_{33} + J2_{33} + J3_{33} + J4_{33} + L1^2 * M2 + L1^2 * M3 + L1^2 * M4 + L2^2 * M3 + L2^2 * M4 + M1 * b1^2 + M2 * b2^2 + 2.0 * L1 * L2 * M3 * cos(q2) + 2.0 * L1 * L2 * M4 * cos(q2) + 2.0 * L1 * M2 * b2 * cos(q2)$

$M_{12} = J2_{33} + J3_{33} + J4_{33} + L2^2 * M3 + L2^2 * M4 + M2 * b2^2 + L1 * L2 * M3 * cos(q2) + L1 * L2 * M4 * cos(q2) + L1 * M2 * b2 * cos(q2)$

$M_{13} = 0$

$M_{14} = J4_{33}$

$M_{22} = J2_{33} + J3_{33} + J4_{33} + L2^2 * M3 + L2^2 * M4 + M2 * b2^2$

$M_{23} = 0$

$M_{24} = J4_{33}$

$M_{33} = M3 + M4$

$M_{34} = 0$

$M_{44} = J4_{33}$

For the matrix of Centrifugal and Coriolis effect based on the Christoffel Symbols we have:

$C_{11} = -1.0 * L1 * dq2 * sin(q2) * (M2 * b2 + L2 * M3 + L2 * M4)$

$C_{12} = -L1 * sin(q2) * (dq1 + dq2) * (M2 * b2 + L2 * M3 + L2 * M4)$

$C_{21} = L1 * dq1 * sin(q2) * (M2 * b2 + L2 * M3 + L2 * M4)$

the rest is zeros.

And the G matrix of gravity effect on the robot:

$G = [0, 0, G * (M3 + M4), 0]'$

By putting forth linkage length and center of mass to zeros, we get the ultimate dynamics of the Conventional SCARA manipulator.

7.1.4.3 Designing Sliding Mode Control for SCARA

In this section we design a sliding mode control for SCARA manipulator for trajectory tracking. for this purpose we assume that external force $\tau_{external}$ is zero. Also friction and gravitational matrix assume be uncertain and bounded as:

$$\|G(q) + \tau_{friction}\| < \beta\sigma \tag{7.1}$$

where β is a positive constant. And the sliding surface σ is given by

$$\sigma = \dot{e}_q - Le_q \tag{7.2}$$

where L is a positive constant. We define position error between joint variables and reference path that found previously as:

$$e_q = q - q_{ref}$$

For dynamic model of SCARA we consider following sliding mode controller:

$$\tau_p = C(q, \dot{q})q_r + M(q)\dot{q}_r - \eta\frac{\sigma}{\|\sigma\|} \tag{7.3}$$

where $\eta > \beta$ is a positive constant and

$$q_r = \dot{q}_d - e$$

. To show that a stable sliding mode exists, we define the Lyapunov function as

$$V = \frac{1}{2}\sigma^T M(q)\sigma \tag{7.4}$$

where $M(q)$ is the inertia matrix and is positive definite. By substituting the control input 7.3 in to the time derivative of V we can obtain:

$$\dot{V} = \sigma^T M(q)\dot{\sigma} + \frac{1}{2}\sigma^T \dot{M}(q)\sigma \tag{7.5}$$

By using Christoffel Symbols which define in chapter 2, we can write:

$$\dot{V} = \sigma^T(M(q)\dot{\sigma} + C(q,\dot{q})\sigma) = \sigma^T(M(q)(\ddot{e}_q + L\dot{e}_q) + C(q,\dot{q})\sigma) \tag{7.6}$$

by using dynamic model of SCARA and inequality 7.1 we can obtain:

$$\begin{aligned}&\dot{V} = \sigma^T(\tau - C(q,\dot{q})\dot{q} - G(q) - \tau_{friction} - M(q)\ddot{q}_d + M(q)\dot{e} + C(q,\dot{q})\sigma) = \\ &\sigma^T(-\eta\tfrac{\sigma}{\|\sigma\|} - G(q) - \tau_f) < -\eta\,\|\sigma\| + \beta\,\|\sigma\| < -\eta\,\|\sigma\|\end{aligned} \tag{7.7}$$

7.1.5 Hybrid Force/Position Control

According to previous sections for position sliding mode control we derived a robust and fast position control. By constructing a Selection function we can make a force control over an unknown surface along the path. If the part of the path is out of object contacting with path line the position controller applied. But when the force goes up to some point predefined, the force controller perpendicular to position control will be applied. The force controller is a PI controller as:

$$\begin{aligned}&\mathrm{u} = \mathrm{Cf(q,dq)} * \mathrm{qr} + \mathrm{Mf(q)} * \mathrm{dqr...} \\ &-\mathrm{etta} * (\mathrm{sig}/(\mathrm{norm(sig)} + 0.1)) - \mathrm{etta} * \mathrm{isig}; \\ &\mathrm{u}(3) = -2000 * \mathrm{ef} + 1000 * \mathrm{ief};\end{aligned}$$

which its coefficients depends on the joint specified for task. In this work by defining a threshold over starting point of force controller, we have reached the ultimate force/position control, which switches between force controller and position controller.

For testing this controller we defined an environment simulation in ADROMS. We assume that the lengthes of the linkages are 1[m] and the masses are 10[kg] and the Inertia s are 0.1[kg.m^2] and the uncertainty bound in equarion 7.1 is defined as $\beta = 100$. First we made a path on the Cylinder with radius of 0.3[m] for a SCARA of DHTablen=[q(1),2,0,1;q(2),0,0,1; 0,-2+q(3),0,0;q(4),0,0,0]. Then made a path planning on it with constrained path planning. For applying, we used a new cylinder with radius of 0.31[m] which is produced by [phix,phiq,Aq,RB]= cntf(x,y,z,q,ko,DHTablen), so the path is 1[cm] beneath the surface of contact with object.

In this function X,Y and Z are symbolic variable of Cartesian space, q is symbolic variable of joint spaces, ko is the number of the objects in space with the robot of Denavit-Hartenberg of DHTable. Outputs of this function is the $\phi(x)$ and $\phi(q)$ and perpendicular vector of Aq to surfaces. RB defines the rigid side of the objects, which if it is -1 then inside or left side of the object is rigid and if 1 the outside or right side of the object is rigid.

without force control robot tries to follow the path but as it is under the surface of the cylinder it forces over the surface and makes lots of contact forces up to 1400[N]s. We define a desired force of sine shape between 10 to 50[N]s and by applying force controller in the SCARA q_3 joint we get results of this:

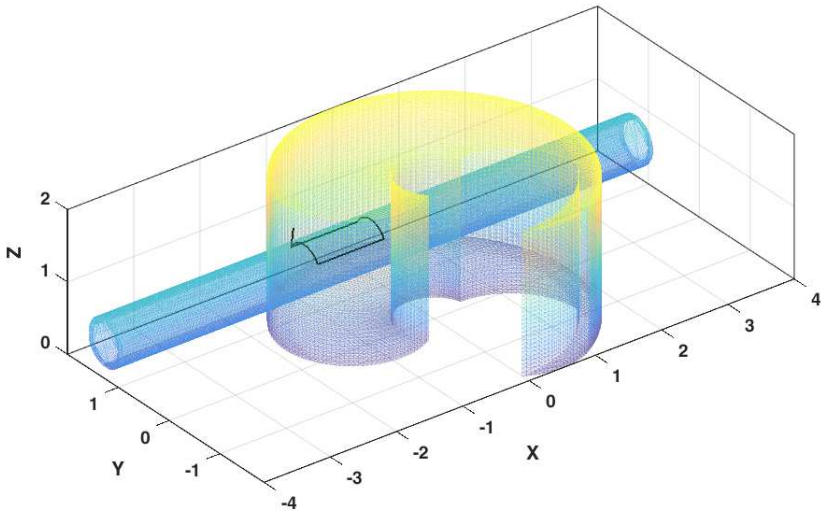


Figure 7.8: Work-Space, Object and the path on it.

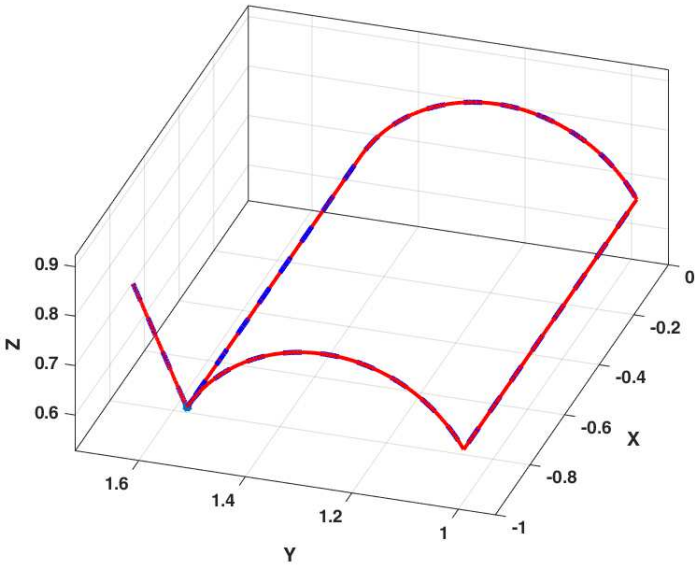


Figure 7.9: Path and its found track.

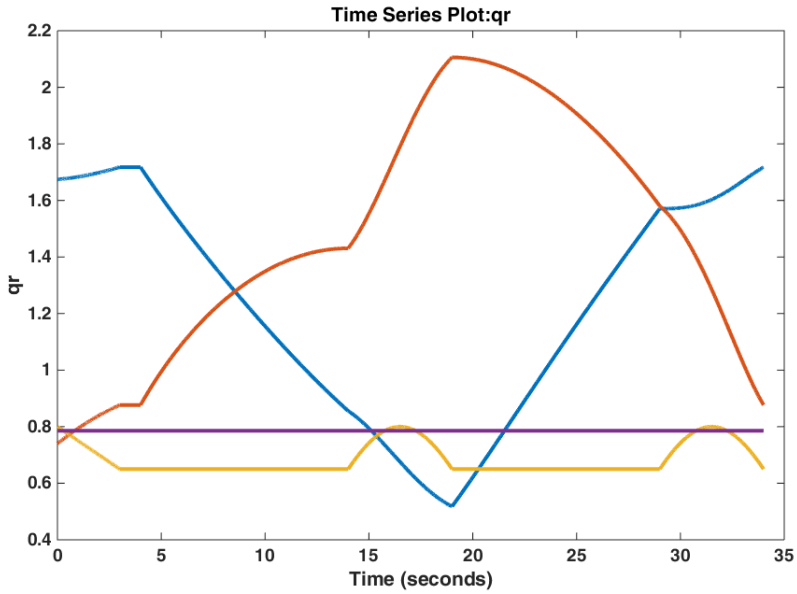


Figure 7.10: Joint space smooth path found.

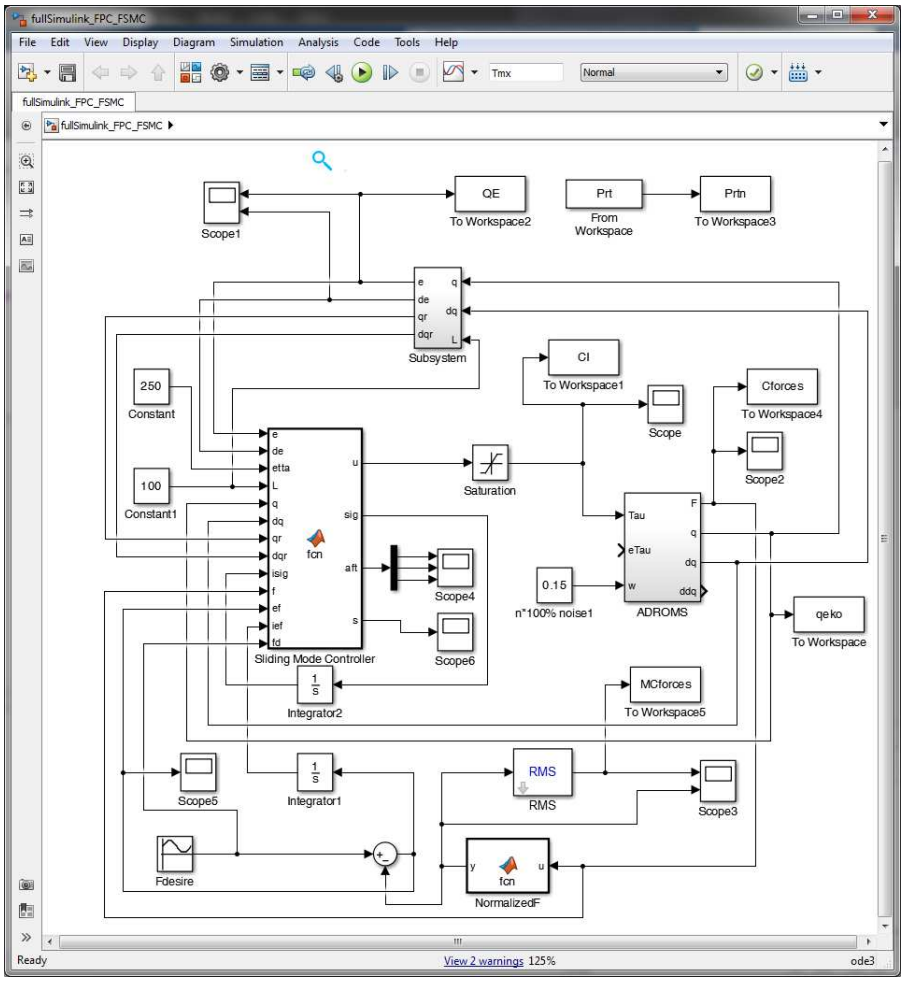


Figure 7.11: Simulink diagram of Hybric Force/Position Control.

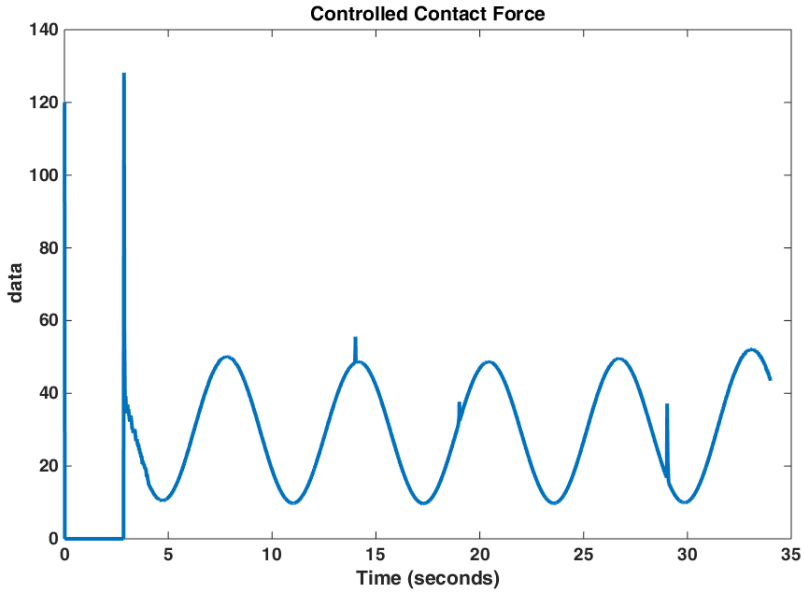


Figure 7.12: Force applied on the surface of cylinder.

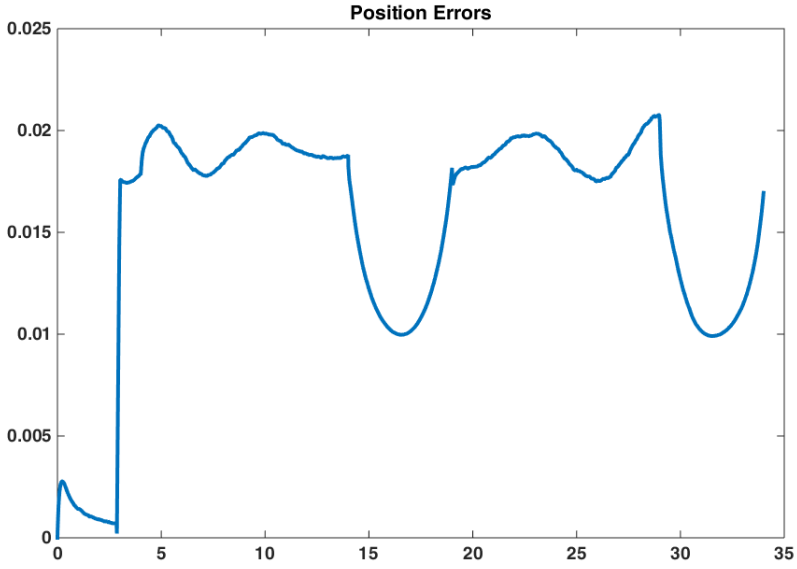


Figure 7.13: Position error of hybric force position.

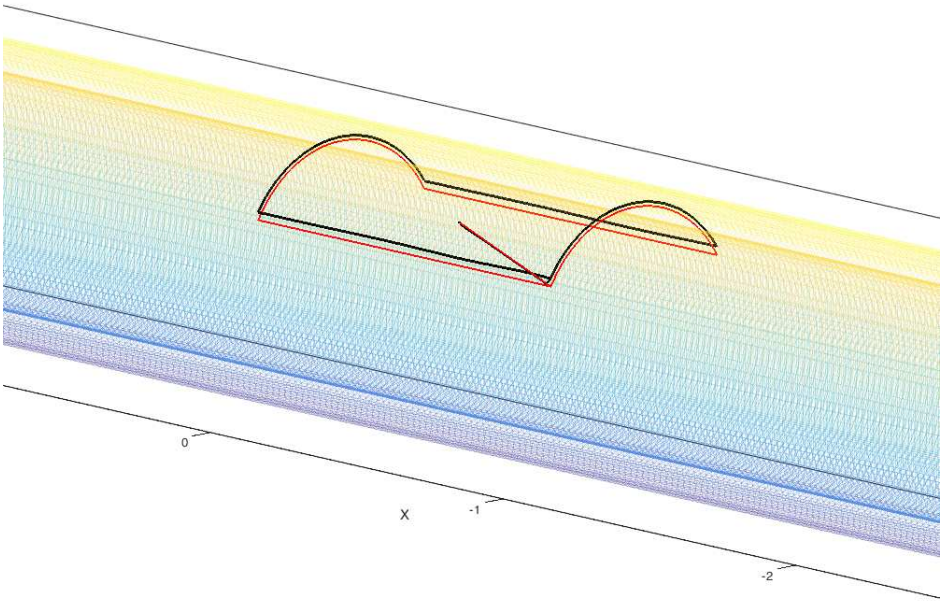


Figure 7.14: Full scheme of the path and guided foce/path position.

Chapter 8

Online Wavelet Complementary Velocity Estimator

8.1 Introduction

High speed in manufacturing of high precision components [217, 218] and micro-feeding mechatronic solutions, is a prime requirement. The main problem of mass production of the precise components is their high price of machinery tools and the time consumption in the production processes. To being precise, it needs high quality in surfacing and dimensioning in the production line. For precise producing goods, todays mechanical structures in analysis and production parts has been reached to the highest levels in decades. But, what mainly effects the precision in the machinery tools, is the vibration induced to the mechanical structures from various sources such as natural frequencies of the working structure in higher speeds and oscillations induced by nonlinear frictions in lower speeds [219]. In heavy duty operations like steel face milling the critical modes for vibration are related to the whole machine tool structure (chatter frequency is between 15 and 100Hz). The introduction of additional damping in the machine tool structure can increase milling stability. However, a passive absorber is not feasible in many machining processes where the dynamics of the system change according to the working position and an active damper is needed. The famous Direct Velocity Feedback (DVF) are well known in active suspension literature [220–223] from old to new methods such as

skyhook [224, 225]. The DVF, which utilizes actuators and the acceleration signal from sensor, is an advanced Mechatronics solution which in performance is comparable with smart materials like piezoelectric actuators in damping strategies. This solution is cheaper in cost, but its reliability is strictly attached to inner algorithms for sensor fusion and controller types. Any good controller has a bandwidth which is limited by its sensory inputs. So using high quality sensor feedback to controller maximizes the whole control system bandwidth. Designing soft sensors to address this issue is an open challenge in literature.

Traditional inertial velocity sensors are based on the movement of a mechanical mass in an electromagnetic field which produces some currents to be sensed. Otherwise piezoelectric velocity sensors are accelerometers with an electronic integrator built in to the unit with a high pass filter for removing the bias created by integrator [226]. So deriving this velocity from online numerical differentiation or integration of a measured data with noises has great importance in signal processing, control engineering [227–229], numerical analysis, or failure diagnostics [230]. It is used in implementation of control strategies which uses the least cost sensors available (just position or angular). But, these low cost sensors deliver such a noise over useful signal which using numerical differentiators over them gives a high noisy results which is practically unusable in high precision works and must use some kinds of filter where each of these filters have their own benefits and problems.

Different online linear filters such as Butterworth or Chebyshev Types I and II, elliptic and some nonlinear filters such as wavelet [231, 232] have been used in the industry for removing this kind of noise from digital differentiators. Another approach is based on the least square on a polynomial structure [233, 234]. High gain observers which adjust the model by weighting the observer output deviations from the system to be controlled [235, 236] is another implementation. Design of numerical differentiators in the frequency domain is based on the assumption that an ideal $n-th$ order differentiator has a frequency response of magnitude ω^n [237–239]. Extended Kalman Filter is a special strategy for state estimation which can be used for digital differentiation [240–242]. The main problem in differentiator design is to combine differentiation exactness with robustness in respect to possible measurement errors and input noises. But most of these filters have delays because of deleting high frequency contents of the signal and phase shifting in much frequencies. If nothing

is known on the signal structure except some differential inequalities, then sliding modes [243, 244] for its exactness and robustness are used. This method is good as the noise on the signal is as low as possible, with high noise the differentiator results contains a high chattering which makes it impossible use in real feedback control systems. A simple and old estimation technique for velocity that is often used in the flight control industry is to fusion measurements in the complementary filter [245]. A complete overview on all these filters has been done in [246].

The Wavelet Transform (WT) is a powerful tool of signal and image processing and it has been vastly used in many scientific areas, such as signal processing, image compression, computer graphics, and pattern recognition [247–251]. Noise filtering using wavelet has been a very mature technology [252–254]. On contrary the traditional Fourier Transform, the WT is particularly suitable for the applications of non-stationary signals which may instantaneous vary in time [255, 256]. But even the wavelet filtering alone on a specific signal, based on its nature tries to remove(or smoothen) the higher resolutions beneath the original signal, so it causes delay in output filtered signal. Many attempts have been done in joining wavelet multi-resolution analysis with kalman filtering to estimate states from high noise data [257–260]. These methods require application of kalman filter over each resolution level, which needs high serially computation resources.

So, finding fast and accurate algorithms which could give the absolute velocity value of the system with low noise, least delay and stable properties, would be a fine interest. This fast and reliable velocity can give us high precision accuracy out of control feedbacks regardless of environmental vibrations.

At this paper, in the next section some preliminaries on discrete wavelet, Butterworth for noise canceling and Kalman filter for state estimation have been introduced. In the third section, structure of the proposed complementary estimator is mathematically explained and proved. The fourth section describes the experimental results in compare to kalman filter, the butterworth and a complementary estimator which uses butterworth filters instead of wavelets.

8.2 Preliminaries

8.2.1 Multi-Resolution Analysis and Discrete Wavelet Transform

Multiresolution Analysis (MRA) is a convenient framework for hierarchical representation of functions or signals on different scales. The basic idea of multiresolution analysis is to represent a function $f(x)$ as a limit of successive approximations. Each of these successive approximations is a smoother version of the original function with more and more of the finer "details" added. Wavelets are terminating basis vectors which are used to decompose a signal using a set of coefficients.

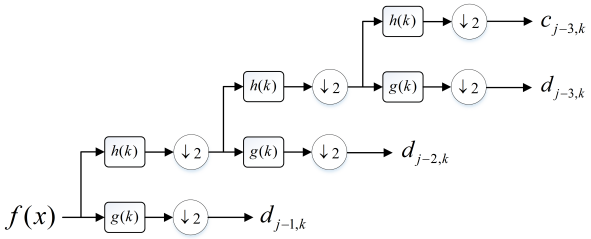


Figure 8.1: Illustration of the analysis part of a three-level decomposition scheme using sub-band coding.

The process of decomposition uses a sub-band coding scheme that is illustrated in Fig.8.1. The Discrete Wavelet Transform (DWT) can be computed using the filter banks $\overline{h(k)}$ and $\overline{g(k)}$ which form a quadrature conjugate mirror filter pair with $h(k)$ and $g(k)$, which are given by Eqs.8.11 from two conjugate functions, the wavelet function $\psi(x)$ and the scaling function $\phi(x)$. The result of the analysis step is a set of intermediate coefficients, which represent the weights of the original signal in terms of the basis functions used, namely the scaling function and the wavelet function. The original sampled signal is filtered with the scaling function and the wavelet function and down sampled by two, resulting in the trend and detail coefficients at level one. The trend coefficients thus obtained are then used as the original signal and filtered with the scaling function and the wavelet to yield the coefficients at level two. This process is repeated depending upon the number of decomposition levels desired. For reverse procedure, the detail and the scale add up, then being

up-sampled and passed from reverse scale filter, ready to being added to detail of the lower levels. By these coefficients the $f(x)$ can be decomposed in wavelet and scale spaces as equations 8.10.

The problems of temporal and frequency resolution found in the analysis of signals with the Short time Fourier Transform (STFT), i.e. best resolution in time at the expense of a lower resolution in frequency and vice-versa, can be reduced through the MRA provided by Discrete Wavelet Transform (DWT). The temporal resolutions, Δt, and frequency, Δf, indicate the precision time and frequency in the analysis of the signal. Both parameters vary in terms of time and frequency, respectively, in signal analysis using DWT. Unlike the STFT, where a higher temporal resolution could be achieved at the expense of frequency resolution. Intuitively, when the analysis is done from the point of view of filters series, the temporal resolution should increase increasing the center frequency of the filters bank. Thus, Δf is proportional to f.

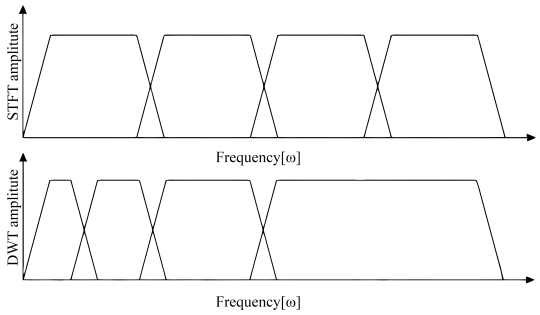


Figure 8.2: Comparison between STFT and DWT in their frequency domain signal decomposition.

The main difference between DWT and STFT is the time-scaling parameter. The result is geometric scaling that gives the DWT logarithmic frequency coverage with nonlinear phase distortion in contrast to the uniform frequency coverage of the STFT, as compared in Fig.8.2.

8.2.2 Butterworth Filter

The Butterworth filter is a type of signal processing filter designed to have as flat a frequency response as possible in the passband. It is also referred to as a maxi-

mally flat magnitude filter. In [261, 262] design procedures has been described. For selection of an optimal cutoff frequency and order has been discussed in [263, 264] exhaustively. But for precision, it has been developed a pattern search optimization method which finds the optimal cutoff frequency and order with comparison of the filter output to offline computed velocity data. These velocity data has been processed by a non casual offline smooth differentiation from position data in 512 sampling windows. The optimal cutoff frequency found for our system is 30$[Hz]$ frequency with the order of 2 by a delay of 7$[ms]$ in the output. In Fig.8.3, use of this filter after a direct differentiator has been shown. As it is obvious there is a delay in results which makes this filter unreliable in high precision-high speed processes.

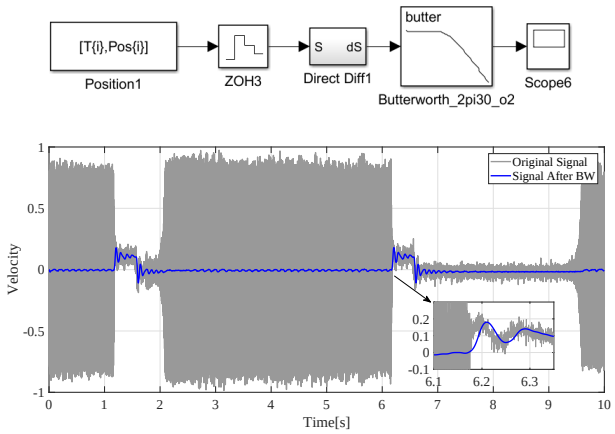


Figure 8.3: Numerical Differentiator followed by a butterworth filter.

8.2.3 Kalman Filter

Kalman filter is the most famous for estimation of system states by fusion of the noisy input signals [265]. In this paper for comparison we have considered a tri-state system differentiator as:

$$\begin{aligned}
\overbrace{\begin{bmatrix} P(k+1) \\ V(k+1) \\ A(k+1) \end{bmatrix}}^{X(k+1)} &= \overbrace{\begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}}^{\Gamma} \overbrace{\begin{bmatrix} P(k) \\ V(k) \\ A(k) \end{bmatrix}}^{X(k)} + w \\
\overbrace{\begin{bmatrix} P_s(k) \\ A_s(k) \end{bmatrix}}^{Y(k)} &= \overbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}^{C} \begin{bmatrix} P(k) \\ V(k) \\ A(k) \end{bmatrix} + v
\end{aligned} \tag{8.1}$$

Where the discrete states are $X(k) = [P(k), V(k), A(k)]'$ and sensed data are $Y(k) = [P_s(k), A_s(k)]'$. The $w = N(0, Q)$ and $v = N(0, R)$ are the uncertainties with Gaussian Distribution of variances Q and R. Kalman filter has a two-step procedure for state estimation. In the prediction step, The current state and error noise covariances are used to project forward through the state model in order to estimate the predicted mean and covariances. This is called as the a priori estimate. Once the outcome of the next measurement (corrupted with random noise) is observed, these estimates are updated using a weighted average, with more weight being given to estimates with higher certainty. These two steps equations for our differentiator case are as follows:

- Initialization: It consists of choosing good initials for X_{k-1}^- and P_{k-1}^-
- Prediction:

$$\begin{aligned}
\hat{X}_k^- &= \Gamma \hat{X}_{k-1} \\
P_k^- &= \Gamma P_{k-1} \Gamma^T + Q
\end{aligned} \tag{8.2}$$

- Update:

$$\begin{aligned}
K_k &= P_k^- C^T (C P_k^- C^T + R)^{-1} \\
\hat{X}_k &= \hat{X}_k^- + K_k (Y - C \hat{X}_k^-) \\
P_k &= (I - K_k C) P_k^-
\end{aligned} \tag{8.3}$$

The algorithm is recursive. It can run in real time, using only the present input measurements and the previously calculated state and its uncertainty matrix, so no additional past information is required.

8.2.4 Complementary Filters

Complementary Filter (CF) scheme is used wildly in many areas of digital technology we have, from aerospace [245,266] to mobile tracking devices [267]. A complementary Filter consists of two or more filters, which in together produce one output from multi-input, where each filter choose to give one part of the output based on the space which the filters are complementary in. Traditionally CFs, shown in Fig.8.4 has been used in finding tilte angle of Mechatronics or UAVs in an accurate low noise signal from accelerometer and gyroscope [268, 269].

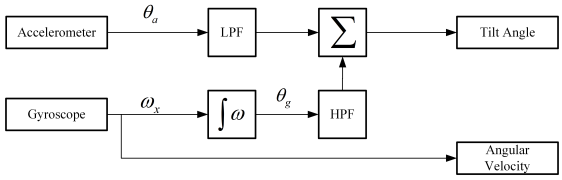


Figure 8.4: Traditional complementary filter for estimating title from accelerometer and gyroscope.

All the complementary filters try to use simple properties of all-pass filters [270]. In an all-pass filter we have two or more filters which the summation of their gain is unity and with their group delay could change the behavior of the signal [271]. Group delay denoted by $\tau_g(\omega)$ vs Phase delay denoted by $\tau_p(\omega)$ is the time delay of the amplitude envelopes of the various sinusoidal components of a signal through a device under test, and is a function of frequency for each component. Phase delay, in contrast, is the time delay of the phase as opposed to the time delay of the amplitude envelope. If the filter's transfer function is $H(j\omega)$, then we have:

$$\begin{aligned} \phi(\omega) &= \arg\{H(j\omega)\} \\ \tau_g(\omega) &= -\frac{d\phi(\omega)}{d\omega} \\ \tau_p(\omega) &= -\frac{\phi(\omega)}{\omega} \end{aligned} \tag{8.4}$$

It is often desirable for the group delay to be constant across all frequencies. In fact, complementary filters are all-pass filters with the lowest flat group delay, which in contrary to all-pass filters, have more than one source of the same signal, but different in each frequency structures. The complementary filter selects the best

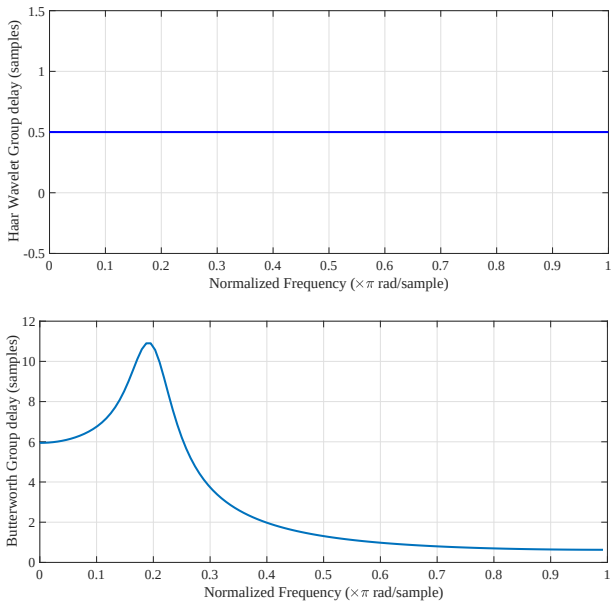


Figure 8.5: Group delays for butterworth and haar wavelet, which shows the haar wavelet has a very low (0.5[Sample]) of group delay.

part of each source with its filters and blend them back to give reliable signal, but with flat group delay specific to sum of all filters group delays. Two group delays of the Haar Wavelet (which we have used) and Butterworth [272] have been shown in Fig.8.5.

8.2.5 Wavelet Transform

Consider a sampled signal $f(x)$ and generate the following sequence of approximations [273]:

$$\begin{aligned} f^m(x) &= \sum_{n=-\infty}^{\infty} f_{m,n}\phi(2^m x - n)_{m=0,1,2,\ldots} \\ f(x) &= \sum_m f^m(x) \end{aligned} \tag{8.5}$$

Each approximation is expressed as the weighted sum of the shifted versions of the same function $\phi(x)$, which is called the scaling function. If the $(m+1)th$ approximation is required to be a refinement of the mth approximation, then the

function $\phi(2^m x)$ should be a linear combination of the basis functions spanning the space of the $(m+1)th$ approximation, i.e.,

$$\phi(2^m x) = \sum_k h(k)\phi(2^{m+1}x - k) \tag{8.6}$$

If V^{m+1} represents the space of all functions spanned by the orthogonal set $\{\phi(2^{m+1}x-k); k \in Z$, the set of integers$\}$ and V^m the space of the coarser functions spanned by the orthogonal set $\{\phi(2^m t - p); p \in Z\}$, then $V^m \subset V^{m+1}$. Let:

$$V^{m+1} = V^m \otimes W^m \tag{8.7}$$

Then W^n is the space that contains the information added upon moving from the coarser $f^m(x)$ to the finer $f^{(m+1)}(x)$ representation of the original function $f(t)$. [273] shows that W^m 's are spaces that are spanned by the orthogonal translates of a single function $\psi(2^m x)$, thus leading to the following equation:

$$f^{(m+1)}(x) = f^m(x) + \sum_{n=-\infty}^{\infty} f_{m,n}\psi(2^m x - n) \tag{8.8}$$

The function $\psi(x)$ is called a wavelet and is related to the scaling function $\phi(x)$, through the following relationship:

$$\psi(2^m x) = \sum_k g(k)\phi(2^{m+1}x - k) \tag{8.9}$$

$h(k)$ and $g(k)$ form a conjugate complementary mirror filter pair. Concluding the discussion, a mixed-form N-level discrete wavelet series representation of the signal is given by:

$$\begin{aligned} f(x) &= \sum_k c_{N,k}\phi_{N,k}(x) + \sum_{m=1}^{N}\sum_k d_{m,k}\psi_{m,k}(x) \\ c_{N,k} &= \sum_k f(x)\overline{\phi_{N,k}(x)} \\ d_{m,k} &= \sum_k f(x)\overline{\psi_{m,k}(x)} \end{aligned} \tag{8.10}$$

where $\overline{\phi(x)}$ and $\overline{\psi(x)}$ are the conjugate functions respectively. Interestingly, the multiresolution concept, besides being intuitive and useful in practice, forms the basis of a mathematical framework for wavelets. One can decompose a function into

a coarse version plus a residual and then iterate this to infinity. If properly done, this can be used to analyze wavelet schemes and derive the wavelet basis. It can be seen from (6) that a wavelet transform decomposes a signal $f(x)$ into trend c and detail d coefficients . An efficient approach in computing the discrete wavelet transform (DWT) is to use the sub-band coding scheme which uses only the filters $h(k)$ and $g(k)$, which are found to be:

$$\begin{aligned} h(k) &= \sqrt{2}\sum_x \phi(x)\overline{\phi(2x-k)} \\ g(k) &= \sqrt{2}\sum_x \psi(x)\overline{\psi(2x-k)} \\ g(k) &= (-1)^k\overline{h(-k+1)} \end{aligned} \tag{8.11}$$

Equations (6) and (7) provide a hierarchical and fast scheme for the computation of the wavelet coefficients of a given function.

The DWT of a signal $f(x)$ results in trend (c) and detail coefficients (d) as given by (6). The first step in signal decomposition consists of computing these trend and detail coefficients. Thereafter, the trend coefficients combined with the scaling function as a basis is used to generate the trend signal [left-hand side of the summation in (6)] and the detail coefficients using the wavelets as a basis are used to generate the detail signals [right-hand side of the summation in (6)]. The trend signal captures the high-scale (low-frequency) information and the detail signal captures the low-scale (high-frequency) information contained in the signal $f(x)$. Depending upon the number of decomposition levels, the end product of a multiresolution decomposition is a set of these signals at different scales (frequencies), as shown in (8), where f_H is the high-scale signal, f_L is the low-scale signal, and $f_{M_i}, i = 1, ..., N-1$, are the medium-scale signals where N is the number of decomposition levels. For example, if a three-level decomposition of error signal is done, it results in one trend signal (low frequency) and three detail signals (high and intermediate frequency). There is redundancy in the trend signal; hence, only one obtained at the last level is chosen. The frequency information of these decomposed signals is approximate since the decomposition process does not use a precise frequency-characterized basis vector such as sines and cosines which are used in Fourier analysis:

$$f(x) = f_H(x) + f_{M_1}(x) + ... + f_{M_{N-1}}(x) + f_L(x) \tag{8.12}$$

8.3 Wavelet Complementary Estimator

As shown in Fig.8.6 and Fig.8.7, We have designed a moving horizon complementary filter in which two wavelet filters produce one output velocity data from two sources. The first source comes from differentiating of position sensor and the second one comes from integrating of acceleration data. If we consider position and acceleration as:

$$\begin{aligned} P(k) &= P_r(\Delta t \times k) + N_p(0, \sigma_p) \\ A(k) &= A_r(\Delta t \times k) + N_a(0, \sigma_a) \end{aligned} \tag{8.13}$$

where $P(k)$ and $A(k)$ are sample of position and acceleration from continuous P_r and A_r with sample time of Δt. These two signals carry white noises with covariances σ_p and σ_a. Now if we take numerical derivative from $P(k)$ and numerical integration from $A(k)$ we get two numerical representation of real velocity $V(n)$ as:

$$\begin{aligned} V_d(n) &= V(n - d_d) + \tfrac{\Delta N(0,\sigma_p)}{\Delta t} \\ V_i(n) &= V(n - d_i) + b_0 n + \textstyle\sum N(0, \sigma_a) \end{aligned} \tag{8.14}$$

where d_d and d_i are the delay produced by numerical operators, differentiation and integration respectively. The first one consists of violet noise (Differentiation of white noise), but without any drift in output and the second one is a good velocity data but having bias b_0 came from numerical integration procedure on initial mean value of acceleration and local mean of discrete white noise and a brown(ian) noise (Integration of white noise). As said in introduction, wavelet filters alone removes data from signals. But here by combining two higher and lower resolutions data from two relevant signal we get almost all data back.

If we consider the delays from operators could be neglected, we can derive the discrete wavelet transform of both velocities as:

$$\begin{aligned} {c^d}_{N,k} &= \sum_k \{V(n) + N_{violet}(0, \sigma)\}\overline{\phi_{N,k}(n)} \\ {d^d}_{m,k} &= \sum_k \{V(n) + N_{violet}(0, \sigma)\}\overline{\psi_{m,k}(n)} \\ {c^i}_{N,k} &= \sum_k \{V(n) + b_0 n + N_{brown}(0, \sigma)\}\overline{\phi_{N,k}(n)} \\ {d^i}_{m,k} &= \sum_k \{V(n) + b_0 n + N_{brown}(0, \sigma)\}\overline{\psi_{m,k}(n)} \end{aligned} \tag{8.15}$$

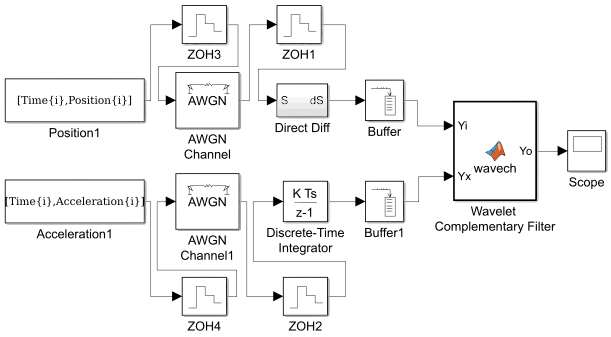


Figure 8.6: Scheme of the Wavelet Complementary Velocity Estimator which is consists of two parts. the upper part is numerical differentiator and the lower part is the numerical integrator.

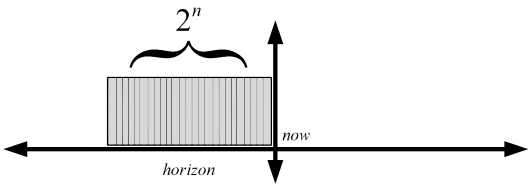


Figure 8.7: Moving Horizon of the Wavelet Complementary Estimator which is consists of 2^n buffered samples.

As the function $\{N_{violet}(0,\sigma)\}\overline{\phi_{N,k}(n)}$ is an even function and $\{b_0 n\}\overline{\psi_{m,k}(n)}$ is an odd function over large k we can expect that the following summations goes to zero:

$$\begin{aligned} C &= \sum_k \{N_{violet}(0,\sigma)\}\overline{\phi_{N,k}(n)} \approx 0 \\ D &= \sum_k \{b_0 n + \overbrace{\overline{N_{brown}(0,\sigma)}}^{\approx 0}\}\overline{\psi_{m,k}(n)} \approx 0 \end{aligned} \tag{8.16}$$

We have constructed our new complementary filter by choosing the filter coefficients as:

$$\begin{aligned} f^c(x) &= \sum_k {c^c}_{N,k}\phi_{N,k}(x) + \sum_{m=1}^{N}\sum_k {d^c}_{m,k}\psi_{m,k}(x) \\ {c^c}_{N,k} &= {c^d}_{N,k} \\ {d^c}_{m,k} &= {d^i}_{m,k} \end{aligned} \tag{8.17}$$

This gives us a results with much less bias and noise. In addition based on the

structure we can connect this estimator concurrently in serially manner.

8.4 Results and Discussions

For illustrating the power of this filter we tried two basket of data. One is the artificial sinusoidal function with white Gaussian noise over them to benchmark it with other methods and the other is some collected position and acceleration data from a real shaking table shown in Fig.8.8. The shaking table is activated by various input voltage signals, where here in this article we have used the step function input as case study. The position and acceleration measurements of the system are carried out by linear potentiometer transducer PC-M-300) and accelerometer (ADXL05EM-EM-3) separately. Every measurement is recorded with the sampling frequency of $2K[Sps]$, which is shown in Fig.8.9.

Figure 8.8: Physical shaking table in our laboratory.

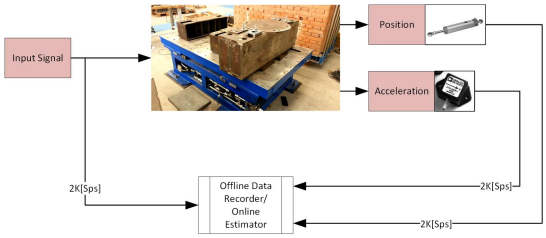


Figure 8.9: Structure of data gathering system.

The sinusoidal function, shown in Fig.8.10 are:

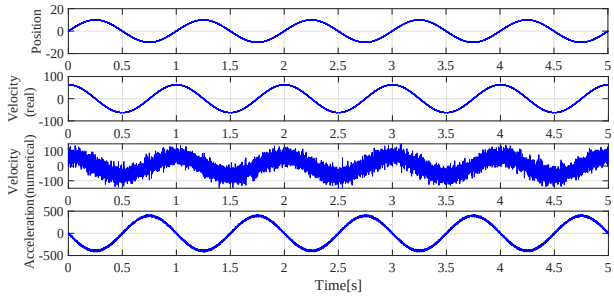


Figure 8.10: Sinusoidal Position, Real Velocity, Velocity from numerical differentiation and Acceleration for benchmarking the Wavelet Complementary Estimator.

$$\begin{cases} S = 10\sin(2\pi t) + N(0, \sigma_1) \\ \frac{dS}{dt} = 20\pi\cos(2\pi t) \\ \frac{d^2S}{dt^2} = -40\pi^2\sin(2\pi t) + N(0, \sigma_2) \end{cases} \tag{8.18}$$

In the simulation case, the position sensor Signal to Noise Ration (SNR) is set to $70[dB]$ and the acceleration sensor SNR is set to $10[dB]$. The Fig.8.11 shows comparison between butterworth and our estimator. The WCE results have almost no delay with respect to butterworth filter in spite of more digestible noise over it. As it is clear the WCE takes more time to converges to the real velocity.

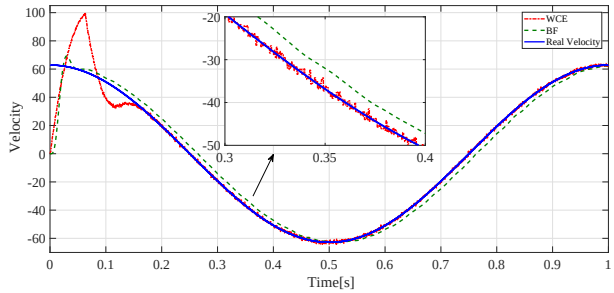


Figure 8.11: Comparison of Wavelet Complementary Estimator with butterworth filter, it shows the delay conventional filters.

The next comparison is with the Kalaman Filter, which has been shown in Fig.8.12. Kalman Filter after 3 seconds has been successfully converged to the real velocity. This result is comparable to next simulation of kalman filter which could

not converge to real velocity, which proposes that the kalman filter is just applicable in data sets which has no bias.

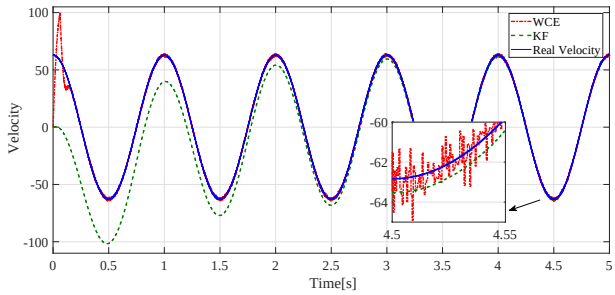


Figure 8.12: Comparison between the Kalman Filter and WCE which shows WCE's faster convergences, but kalman filter's better results in non biased data sets.

Besides our complementary estimator, we have designed an Ordinary Complementary Estimator (OCE) that uses traditional butterworth filters, one highpass for integration part and one lowpass for differentiation part, instead of our wavelet filter banks. The results are shown in Fig.8.13.

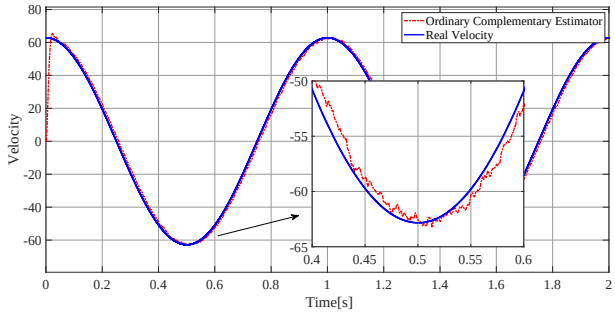


Figure 8.13: Results of the ordinary complementary estimator shows a stable, fast converging but biased output.

For benchmarking the WCE, we have considered the long term ($10[s]$) velocity integration to compare it with real position. This test bench shows how much the data produced can be trustable for producing stable control outputs in feedback systems. Fig.8.14 shows the integration results. Integration of the Kalman Filter goes wild off the road, but butterworth and the WCE stays in stable manner. As it is obvious, WCE has no delay with compare to the butterworth filter.

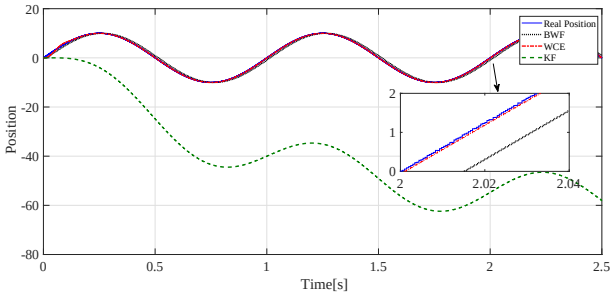


Figure 8.14: Integration benchmark for Kalman and Butterworth filters and the WCE. Kalman Filter goes off and the other two stays with real position data.

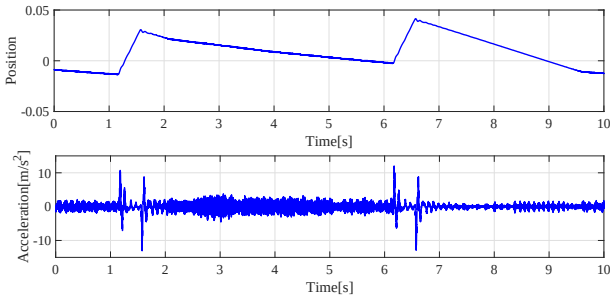


Figure 8.15: Position and Acceleration gathered from sensors.

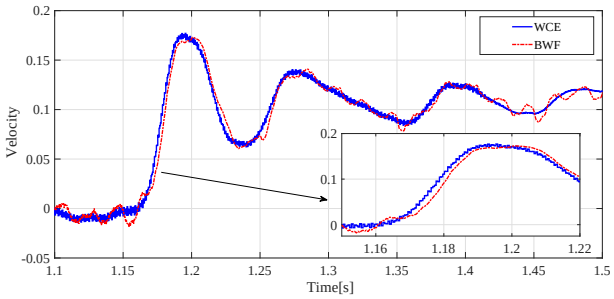


Figure 8.16: Velocity estimation by Wavelet Complementary and Butterworth $2\pi 60(rad) order 2$ Filters.

For better comparison the Mean Square Error of the results has been tabled in the table.8.1.

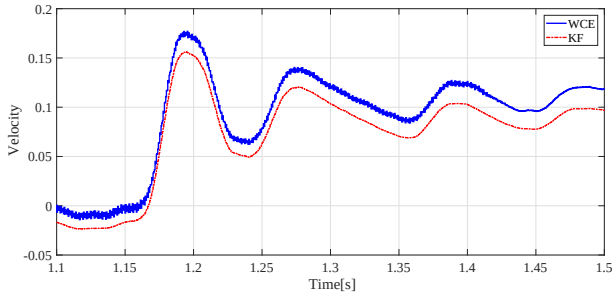


Figure 8.17: Velocity estimation by Wavelet Complementary and Kalman filters.

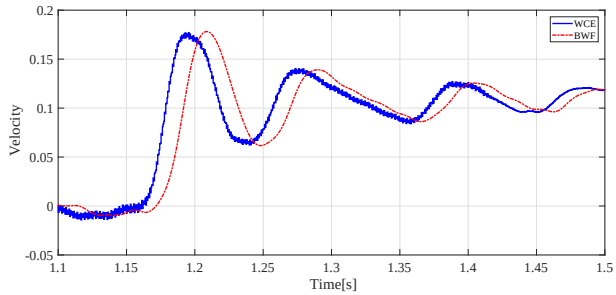


Figure 8.18: Comparing velocity estimation with Butterworth filtering $2\pi 30(rad) order 4$ after numerical differentiator.

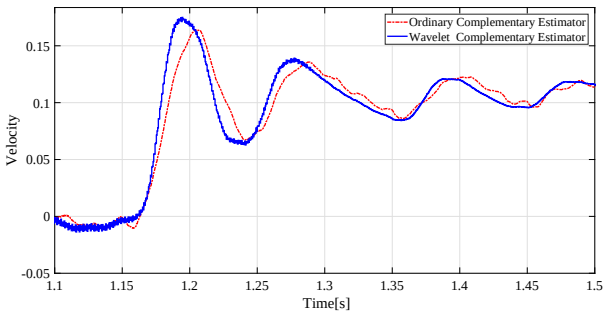


Figure 8.19: Velocity estimation by Wavelet Complementary and Ordinary Complementary with butterworth $2\pi 30(rad) order 2$ Filters.

In the Fig.8.15, a gathered data set of the position and acceleration is shown. We try to estimate the velocity of system based on these two gathered data.

In a control feedback, usually we need a stable and clean data to compensate

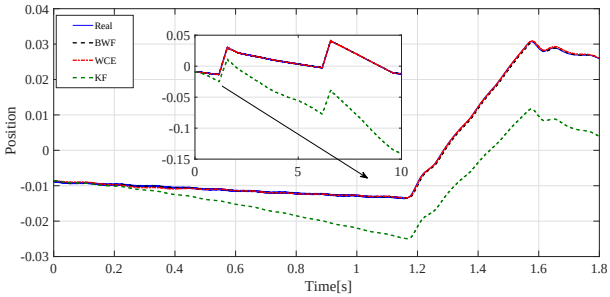


Figure 8.20: Position from velocity integration of filters to comparison, as seen kalman filter has a large bias in this benchmark.

Table 8.1: Mean Square Error comparison of four filters in Velocity and Position from long integration for sinusoidal signal.

	WCE	KalmanF	BWF	OCE
MSE[Vel]	3.629	100.003	1.974×10^3	5.925
MSE[IPos]	0.002	3.701×10^3	0.3314	0.126

the disturbances in output. So, removing delay and stabilization are the most important goals of any state estimators. Here in Figs.8.16-8.19, we have compared the velocity results of the four methods, Butterworths, Kalman Filter and Wavelet Complementary Estimator with real data. The butterworth filter has order 2 with cutoff frequency of $60[Hz]$. The moving horizon width of the Wavelet Complementary Estimator is chosen 256 samples with the Haar wavelet kernel. The results of Fig.8.16 shows that the butterworth filter even with high cutoff frequency has the delay and oscillation problems. In contrary kalman filter results are much smooth and stable, but with a large bias that is unusable in state feedback nonlinear controllers. For illustration of the stability of the WCE, in Fig.8.20 the integration of these three filters has been shown which implies the correctness of the WCE beside the conventional buttherworth.

8.5 Conclusions

In this chapter, we have proposed a simple algorithm to estimate velocity from position and acceleration data for active vibration control of high speed high preci-

sion processes. As seen, The proposed Wavelet Complementary velocity Estimator has the best stability with lowest delay results with respect to butterworth filtering methods and kalman filter estimating. This low delay comes from its low total group delay with respect to butterworth filter. But the mse results in table.8.1 show that the ordinary complementary estimator (OCE) with traditional butterworth has almost similar but with much delay, which provides another scheme for designing whether if such delay is acceptable. Because of its low delay and integration stability, it is much more suitable for using in the online control feedback applications. The main problem for WCE, is the noise in numerical differentiation part. As the noise goes up for finding the low resolution of the filter we have to take more samples as moving windows horizon, which causes more deviation of results from real velocity. This problem can be resolved with considering better numerical differentiator instead of our crude one step type. As its nature, the algorithm is suitable for parallelism and can be implemented on an FPGA. The MATLAB simulation and data can be downloaded from:

https://www.mathworks.com/matlabcentral/
fileexchange/62831-real-time-
wavelet-complementary-velocity-estimator

This simulation reconstruct the online situation in which the algorithm has been established.

Chapter 9

Object position estimation from optical flow

Object tracking is a very popular research topic in the computer vision field because of its wide applications in video surveillance systems, intelligent driver assistant systems, robot vision, etc [274, 275]. Many algorithms to extract relative velocity from moving objects optical flow has been developed and evaluated [276–279]. Most of these algorithms have been focused usage in outdoor environments in an uncontrolled areas. But in an industrial environment we have access the knowledge over many aspects of that area. This gives us an advantage for designing more specified rational algorithms. In this chapter, we will present our new algorithm for extracting exact velocity based on the real time feature extraction from specified regions of the image. In fact we have used two different aspects of the video processing. We have a higher conclusive computation which extract abstract structures from picture and a very fast feature extraction and new simple optical flow estimation. abstract extractor(In this project, red color detector) finds the regions where our fast feature extractor should operate on. The data hierarchy of our algorithm has been illustrated in figure.9.1. As seen, the data which captured from camera goes into processing up until the estimated position and velocity would be available. But, this data hierarchy just show the data structure of the program. The amount of the processing needed to this method is that much high that if all of these were implemented in a serial manner it would take at least one minute to compute all.

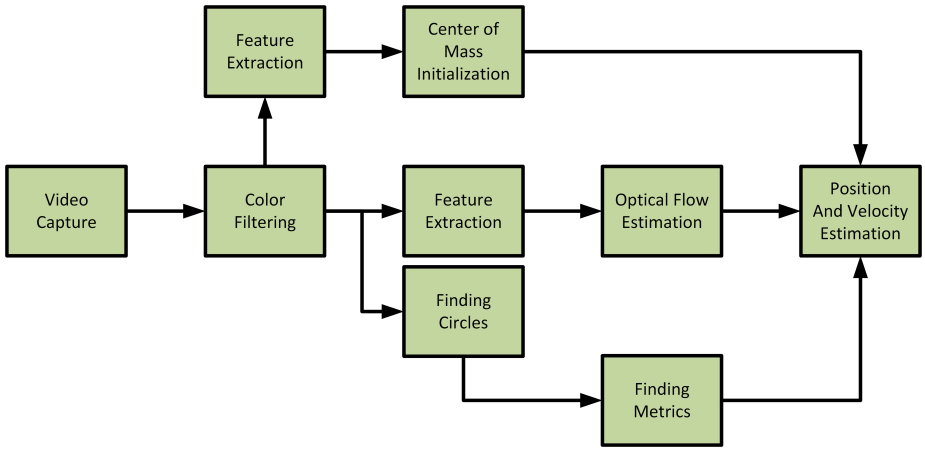


Figure 9.1: Data flow from camera to position estimation.

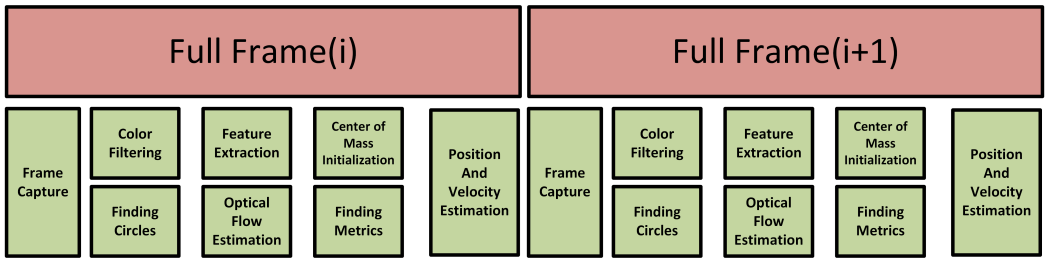


Figure 9.2: Parallel time table of velocity estimator.

So, we have designed an optimized version of these algorithm in a parallel mode. The time table of this algorithm has been shown in figure.9.2. In this manner all the sub algorithms works in a parallel way and as the frame capturing from the camera has the highest time consumption 20$[ms]$, then our program can operate in 50$[Hz]$ at least.

9.1 Results and discussions

For evaluating our method we tested on a mechanical shaking table named in previous chapter. The test rig structure has been sketched in figure.9.3. As it is shown, the test bench includes, our camera (BlueFox-IGC) in specific position, a test paper and shaking table equipments. The shaking table consists of table and hydraulic jack

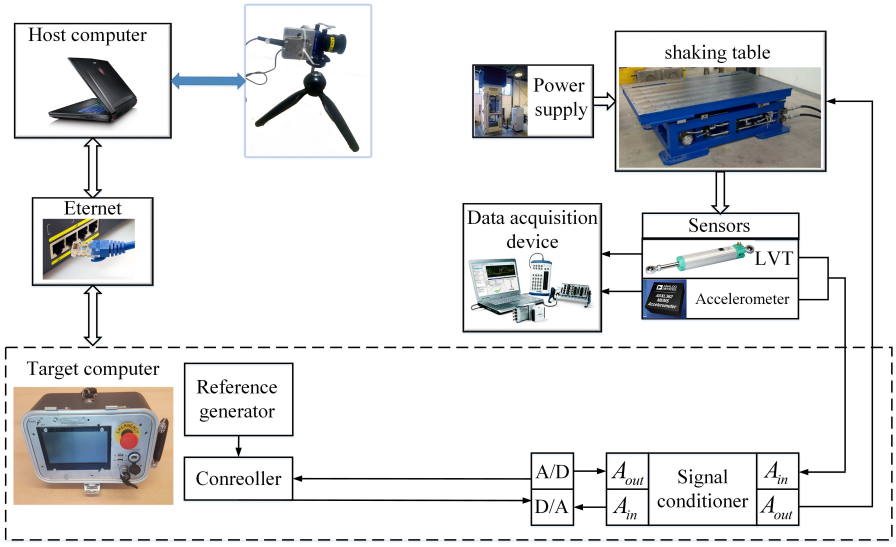


Figure 9.3: Schematic drawing of experimental setup including measurement , control devices and the camera.

which its position is controlled by an advanced controller, and its data is gathered through a national instrument ADC. We have tested sinusoidal waves from $1[hz]$ to $8[Hz]$ of frequencies as position output of shaking table. This tested positions are saved through our camera with $50[fps]$ and saved with MATLAB image acquisition toolbox to test over them later.

9.1.1 Procedure structure

The procedure starts by initializing variables to zero state, because of using all variable in advanced pointer format. Then our first frame capture starts mean the while of the other threads.

As shown in figure.9.1, data directly goes to color filtering. In this part the area of the picture whose color is red will be distinguished and a sparse mask with be produced. This mask will directly be injected to feature extraction part.

The feature extraction part takes the same frame at the time which color filter take it. So, the mask which feature extractor uses always is one frame delay with respect to real position of red objects. We assume the red objects velocities are not so much higher than the circles pixels radius. So this mask just says to feature

extractor to search almost the area which is needed. This makes our algorithm very fast and accurate.

The extracted features will be examined to find the optical velocity which will estimate the next position of the feature(which in our case estimates the now position of object). By differing this position from next frame feature we can find the velocity of one feature. But as we have requested high number of feature to be estimated of one object, so we can assume that these velocities mean values shows our real objects position displacement.

The extracted displacement will be added to the previous stable position(red dot) and estimates the position of the object. In the next section we will see how much accurate this could be.

Mean while, color mask extracted from firt part will be given to metric finder function in parallel which from 4 greatest circle calculates the pixel to mm ratio of the frame. We have a circle of 19mm diameter, so this ratio will be 9.5 in pixels.

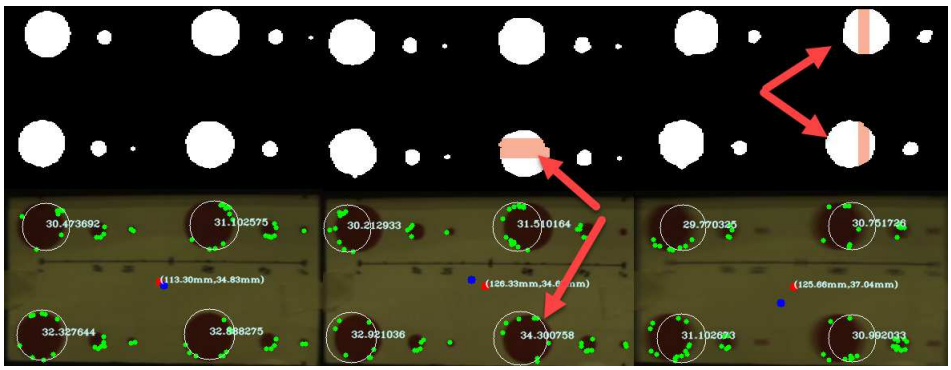


Figure 9.4: Frames grabbed to investigate the algorithm in practice. From left to right we have 1$[Hz]$, 3$[Hz]$ and 8$[Hz]$. In the middle and last frame the delay in color extraction can be completely obvious, but the green dots which are from feature extraction are clear. The tears in color eaxtraction can be seen in orange color with red arrows.

9.1.2 Video Analysis

In figures.9.4, video captures and illustrations of sinusoidal waves for 1,2 and 3$[Hz]$ has been shown. The red dot is our origin of moving object, which has been initial-

ized at first frame by 2000 features extractions from red filtered areas. Now on, this position has been updated by addition of displacement estimated from averaging of 64 features extraction of each frame at once.

As our program tries to fetch data in the fastest mode, the processing may occur between changing of two frames data swapping, leading to make tears in frames. This tear makes the processing output in some cases invalid, in this case the metric extraction at that moment.

9.1.3 Position Extraction

After saving all the data needed, we can visualize how much our estimation goes. In following figures we can see the estimated positions for these three frequencies. As the frequency goes up the results goes more vaguely showing real time positions as their low fps.

Based on the experiments, we have concluded that this method can extract positions with frequencies up to $5[Hz]$ with very high(sub millimeter) accuracies. This allows us to make real time velocity estimations by fusion these positions with help of our previous WCE velocity estimator with help of real time accelerations.

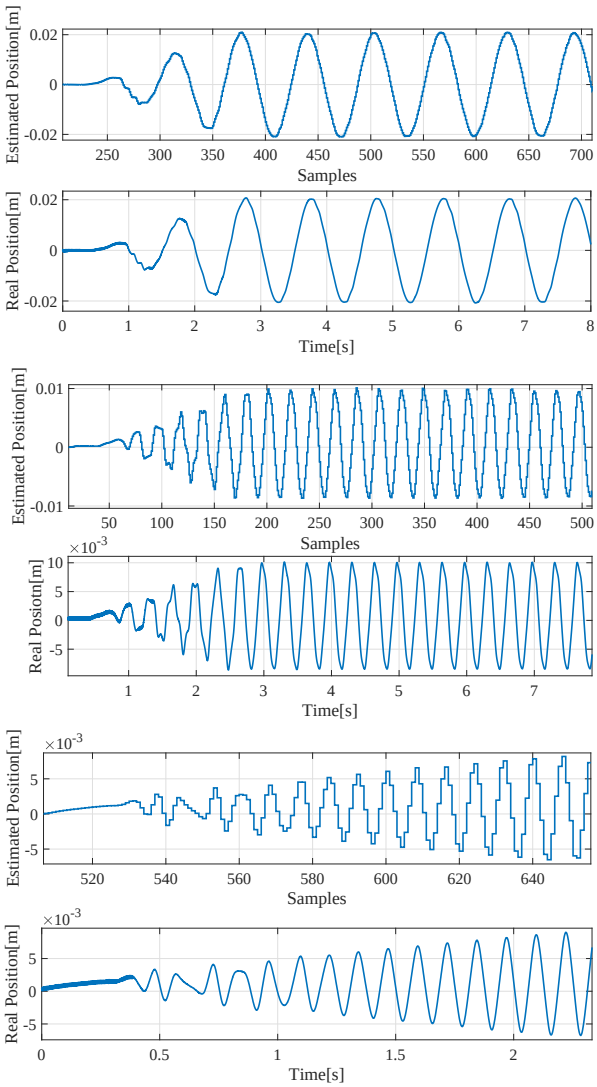


Figure 9.5: From up to down we have $1[Hz]$, $3[Hz]$ and the $8[Hz]$. As seen all the cures shows very accurate position estimation for our algorithms.

Part IV

References

Bibliography

[1] G. Hegde, *A Texbook of Industrial Robotics*. Laxmi Publications, 2006.

[2] J. Pires, A. Loureiro, and G. Bölmsjo, *Welding Robots: Technology, System Issues and Application*. Springer London, 2006.

[3] Wikipedia, "Remote center compliance," tech. rep., 2017.

[4] Y. ling Yang, Y. ding Wei, J. qiang Lou, L. Fu, and X. wei Zhao, "Nonlinear dynamic analysis and optimal trajectory planning of a high-speed macro-micro manipulator," *Journal of Sound and Vibration*, vol. 405, no. Supplement C, pp. 112 – 132, 2017.

[5] E. Zalama, P. Gaudiano, and J. L. Coronado, "A real-time, unsupervised neural network for the low-level control of a mobile robot in a nonstationary environment," *Neural networks*, vol. 8, no. 1, pp. 103–123, 1995.

[6] E. Magrini, F. Flacco, and A. De Luca, "Control of generalized contact motion and force in physical human-robot interaction," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pp. 2298–2304, IEEE, 2015.

[7] S. Part, "Impedance control: An approach to manipulation," *Journal of dynamic systems, measurement, and control*, vol. 107, p. 17, 1985.

[8] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 102, no. 127, pp. 126–133, 1981.

[9] W. Gao, S. Kim, H. Bosse, H. Haitjema, Y. Chen, X. Lu, W. Knapp, A. Weckenmann, W. Estler, and H. Kunzmann, "Measurement technologies for precision positioning," *CIRP Annals*, vol. 64, no. 2, pp. 773 – 796, 2015.

[10] M. Gilles, H. Fayad, P. Miglierini, J. Clement, S. Scheib, L. Cozzi, J. Bert, N. Boussion, U. Schick, O. Pradier, *et al.*, "Patient positioning in radiotherapy based on surface imaging using time of flight cameras," *Medical physics*, vol. 43, no. 8, pp. 4833–4841, 2016.

[11] K. Wang, Y. Liu, and L. Li, "Visual servoing trajectory tracking of nonholonomic mobile robots without direct position measurement," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 1026–1035, 2014.

[12] C. E. Reiley, T. Akinbiyi, D. Burschka, D. C. Chang, A. M. Okamura, and D. D. Yuh, "Effects of visual force feedback on robot-assisted surgical task performance," *The Journal of thoracic and cardiovascular surgery*, vol. 135, no. 1, pp. 196–202, 2008.

[13] A. Kawamura, K. Tahara, R. Kurazume, and T. Hasegawa, "Robust manipulation for temporary lack of sensory information by a multi-fingered hand-arm system," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pp. 4201–4206, IEEE, 2011.

[14] M. H. Lee and H. R. Nicholls, "Review article tactile sensing for mechatronics—a state of the art survey," *Mechatronics*, vol. 9, no. 1, pp. 1–31, 1999.

[15] H. Yousef, M. Boukallel, and K. Althoefer, "Tactile sensing for dexterous in-hand manipulation in robotics—a review," *Sensors and Actuators A: physical*, vol. 167, no. 2, pp. 171–187, 2011.

[16] Z. Kappassov, J.-A. Corrales, and V. Perdereau, "Tactile sensing in dexterous robot hands," *Robotics and Autonomous Systems*, vol. 74, pp. 195–220, 2015.

[17] M. Fallon, S. Kuindersma, S. Karumanchi, M. Antone, T. Schneider, H. Dai, C. P. D'Arpino, R. Deits, M. DiCicco, D. Fourie, *et al.*, "An architecture for online affordance-based perception and whole-body planning," *Journal of Field Robotics*, vol. 32, no. 2, pp. 229–254, 2015.

[18] M. R. Kounte and D. B. Sujatha, "A review of modelling visual attention using computational cognitive neuroscience for machine vision," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 9, pp. 3558–3563, 2013.

[19] F. Chaumette, S. Hutchinson, and P. Corke, "Visual servoing," in *Springer Handbook of Robotics*, pp. 841–866, Springer, 2016.

[20] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review," *IEEE transactions on cybernetics*, vol. 43, no. 5, pp. 1318–1334, 2013.

[21] A. A. Shetty, C. G. Nayak, and V. George, "Stereo vision for mobile robots: A review," *International Journal of Engineering & Technology*, vol. 2, no. 11, pp. 25–29, 2015.

[22] G. Podrekar, B. Bratanič, B. Likar, F. Pernuš, and D. Tomaževič, "Automated visual inspection of pharmaceutical tablets in heavily cluttered dynamic environments," in *Machine Vision Applications (MVA), 2015 14th IAPR International Conference on*, pp. 206–209, IEEE, 2015.

[23] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *IEEE Robotics & Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.

[24] W. Jang and Z. Bien, "Feature-based visual servoing of an eye-in-hand robot with improved tracking performance," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pp. 2254–2260, IEEE, 1991.

[25] P. Marcoux, *Fine pitch surface mount technology: quality, design, and manufacturing techniques*. Springer Science & Business Media, 2013.

[26] G. J. Agin, *Real time control of a robot with a mobile camera*. SRI International, 1979.

[27] L. Acosta, J. Rodrigo, J. A. Mendez, G. N. Marichal, and M. Sigut, "Ping-pong player prototype," *IEEE robotics & automation magazine*, vol. 10, no. 4, pp. 44–52, 2003.

[28] R. L. Andersson, "Aggressive trajectory generator for a robot ping-pong player," *IEEE Control Systems Magazine*, vol. 9, no. 2, pp. 15–21, 1989.

[29] E. P. Krotkov, *Active computer vision by cooperative focus and stereo*. Springer Science & Business Media, 2012.

[30] J. R. Flanagan, M. C. Bowman, and R. S. Johansson, "Control strategies in object manipulation tasks," *Current opinion in neurobiology*, vol. 16, no. 6, pp. 650–659, 2006.

[31] J. F. Seara and G. Schmidt, "Intelligent gaze control for vision-guided humanoid walking: methodological aspects," *Robotics and Autonomous Systems*, vol. 48, no. 4, pp. 231–248, 2004.

[32] Z. Pan, J. Polden, N. Larkin, S. Van Duin, and J. Norrish, "Recent progress on programming methods for industrial robots," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 87–94, 2012.

[33] W. S. Newman and J. J. Patel, "Experiments in torque control of the adeptone robot," in *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pp. 1867–1872 vol.2, Apr 1991.

[34] I. F. of Robotics, "Executive summary world robotics 2016 service robots," tech. rep., International Federation of Robotics, 2016.

[35] Z. Qu, "Global stability of trajectory tracking of robot under pd control," *Dynamics and Control*, vol. 4, no. 1, pp. 59–71, 1994.

[36] R. Kelly, "Pd control with desired gravity compensation of robotic manipulators a review," *The International Journal of Robotics Research*, vol. 16, no. 5, pp. 660–672, 1997.

[37] Q. Chen, H. Chen, Y. Wang, and P. Woo, "Global stability analysis for some trajectory-tracking control schemes of robotic manipulators," in *American Control Conference, 2000. Proceedings of the 2000*, vol. 5, pp. 3343–3347, IEEE, 2000.

[38] E. M. Jafarov, M. A. Parlakçi, and Y. Istefanopulos, "A new variable structure pid-controller design for robot manipulators," *Control Systems Technology, IEEE Transactions on*, vol. 13, no. 1, pp. 122–130, 2005.

[39] J. Choi and J. Lee, "Adaptive iterative learning control of uncertain robotic systems," *IEE Proceedings-Control Theory and Applications*, vol. 147, no. 2, pp. 217–223, 2000.

[40] J. Perk, G.-S. Han, H.-S. Ahn, and D.-H. Kim, "Adaptive approaches on the sliding mode control of robot manipulators," *Transactions on Control, Automation and Systems Engineering*, vol. 3, no. 1, pp. 15–20, 2001.

[41] C.-C. Cheah, C. Liu, and J.-J. E. Slotine, "Adaptive tracking control for robots with unknown kinematic and dynamic properties," *The International Journal of Robotics Research*, vol. 25, no. 3, pp. 283–296, 2006.

[42] Y. Li, S. Tong, and T. Li, "Adaptive fuzzy output feedback control for a single-link flexible robot manipulator driven dc motor via backstepping," *Nonlinear Analysis: Real World Applications*, vol. 14, no. 1, pp. 483–494, 2013.

[43] L. Tang, Y.-J. Liu, and S. Tong, "Adaptive neural control using reinforcement learning for a class of robot manipulator," *Neural Computing and Applications*, vol. 25, no. 1, pp. 135–141, 2014.

[44] V. Pilania and K. Gupta, "A hierarchical and adaptive mobile manipulator planner with base pose uncertainty," *Autonomous Robots*, pp. 1–21, 2015.

[45] P. Poignet and M. Gautier, "Nonlinear model predictive control of a robot manipulator," in *Advanced Motion Control, 2000. Proceedings. 6th International Workshop on*, pp. 401–406, IEEE, 2000.

[46] T. Henmi, T. Ohta, M. Deng, and A. Inoue, "Tracking control of a two-link planar manipulator using nonlinear model predictive control," *International Journal of Innovative Computing, Information and Control*, vol. 6, no. 7, pp. 2977–2984, 2010.

[47] S. K. Pradhan and B. Subudhi, "Nonlinear adaptive model predictive controller for a flexible manipulator: An experimental study," *IEEE Transactions on Control Systems Technology*, vol. 22, pp. 1754–1768, 2014.

[48] M. Khairudin, Z. Mohamed, and A. R. Husain, "Dynamic model and robust control of flexible link robot manipulator," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 9, no. 2, pp. 279–286, 2013.

[49] R.-J. Wai and P.-C. Chen, "Robust neural-fuzzy-network control for robot manipulator including actuator dynamics," *Industrial Electronics, IEEE Transactions on*, vol. 53, no. 4, pp. 1328–1349, 2006.

[50] W. Shang and S. Cong, "Robust nonlinear control of a planar 2-dof parallel manipulator with redundant actuation," *Robotics and Computer-Integrated Manufacturing*, vol. 30, no. 6, pp. 597–604, 2014.

[51] G. Bartolini, A. Pisano, E. Punta, and E. Usai, "A survey of applications of second-order sliding mode control to mechanical systems," *International Journal of Control*, vol. 76, no. 9-10, pp. 875–892, 2003.

[52] W.-H. Chen, "Disturbance observer based control for nonlinear systems," *Mechatronics, IEEE/ASME Transactions on*, vol. 9, no. 4, pp. 706–710, 2004.

[53] A. Calanca, L. M. Capisani, A. Ferrara, and L. Magnani, "An inverse dynamics-based discrete-time sliding mode controller for robot manipulators," in *Robot Motion and Control 2007*, pp. 137–146, Springer, 2007.

[54] A. Ferrara and L. Magnani, "Motion control of rigid robot manipulators via first and second order sliding modes," *Journal of Intelligent and Robotic Systems*, vol. 48, no. 1, pp. 23–36, 2007.

[55] L. M. Capisani, A. Ferrara, and L. Magnani, "Design and experimental validation of a second-order sliding-mode motion controller for robot manipulators," *International Journal of Control*, vol. 82, no. 2, pp. 365–377, 2009.

[56] S. Islam and X. P. Liu, "Robust sliding mode control for robot manipulators," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 6, pp. 2444–2453, 2011.

[57] M.-S. Kang, "Disturbance observer based sliding mode control for link of manipulator driven by elastic cable," *Transactions of the Korean Society for Noise and Vibration Engineering*, vol. 22, no. 10, pp. 949–958, 2012.

[58] V. Venkatesan, S. Mohan, and J. Kim, "Disturbance observer based terminal sliding mode control of an underwater manipulator," in *Control Automation Robotics & Vision (ICARCV), 2014 13th International Conference on*, pp. 1566–1572, IEEE, 2014.

[59] T. Van Tran, Y. Wang, H. Ao, and T. K. Truong, "Sliding mode control based on chemical reaction optimization and radial basis functional link net for deicing robot manipulator," *Journal of Dynamic Systems, Measurement, and Control*, vol. 137, no. 5, p. 051009, 2015.

[60] C. Scherer and S. Weiland, "Linear matrix inequalities in control," *Lecture Notes, Dutch Institute for Systems and Control, Delft, The Netherlands*, 2000.

[61] Y. Wang, L. Xie, and C. E. de Souza, "Robust control of a class of uncertain nonlinear systems," *Systems & Control Letters*, vol. 19, no. 2, pp. 139–149, 1992.

[62] S. P. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear matrix inequalities in system and control theory*, vol. 15. SIAM, 1994.

[63] H. H. Choi, "Output feedback variable structure control design with an h2 performance bound constraint," *Automatica*, vol. 44, no. 9, pp. 2403–2408, 2008.

[64] T.-H. S. Li and Y.-C. Huang, "Mimo adaptive fuzzy terminal sliding-mode controller for robotic manipulators," *Information Sciences*, vol. 180, no. 23, pp. 4641–4660, 2010.

[65] J. K. Salisbury, "Active stiffness control of a manipulator in cartesian coordinates," in *Decision and Control including the Symposium on Adaptive Processes, 1980 19th IEEE Conference on*, pp. 95–100, IEEE, 1980.

[66] C.-h. Wu and R. P. Paul, "Manipulator compliance based on joint torque control," in *Decision and Control including the Symposium on Adaptive Processes, 1980 19th IEEE Conference on*, pp. 88–94, IEEE, 1980.

[67] M. T. Mason, "Compliance and force control for computer controlled manipulators," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 11, no. 6, pp. 418–432, 1981.

[68] J. K. Salisbury and J. J. Craig, "Articulated hands force control and kinematic issues," *The International journal of Robotics research*, vol. 1, no. 1, pp. 4–17, 1982.

[69] C.-h. Wu, "Compliance control of a robot manipulator based on joint torque servo," *The International journal of robotics research*, vol. 4, no. 3, pp. 55–71, 1985.

[70] D. E. Whitney, "Historical perspective and state of the art in robot force control," *The International Journal of Robotics Research*, vol. 6, no. 1, pp. 3–14, 1987.

[71] H. Ishikawa, C. Sawada, K. Kawasa, and M. Takata, "Stable compliance control and its implementation for a 6 dof manipulator," in *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pp. 98–103, IEEE, 1989.

[72] G. Lebret, K. Liu, and F. L. Lewis, "Dynamic analysis and control of a stewart platform manipulator," *Journal of Robotic systems*, vol. 10, no. 5, pp. 629–655, 1993.

[73] Y.-T. Lee, H.-R. Choi, W.-K. Chung, and Y. Youm, "Stiffness control of a coupled tendon-driven robot hand," *Control Systems, IEEE*, vol. 14, no. 5, pp. 10–19, 1994.

[74] E. Dégoulange and P. Dauchez, "External force control of an industrial puma 560 robot," *Journal of robotic systems*, vol. 11, no. 6, pp. 523–540, 1994.

[75] I. Kao, M. R. Cutkosky, and R. S. Johansson, "Robotic stiffness control and calibration as applied to human grasping tasks," *Robotics and Automation, IEEE Transactions on*, vol. 13, no. 4, pp. 557–566, 1997.

[76] Y. Li and I. Kao, "A review of modeling of soft-contact fingers and stiffness control for dextrous manipulation in robotics," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 3, pp. 3055–3060, IEEE, 2001.

[77] S.-F. Chen, Y. Li, and I. Kao, "A new theory in stiffness control for dextrous manipulation," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 3, pp. 3047–3054, IEEE, 2001.

[78] Y. Li, S.-F. Chen, and I. Kao, "Stiffness control and transformation for robotic systems with coordinate and non-coordinate bases," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 1, pp. 550–555, IEEE, 2002.

[79] A. Muller, "Stiffness control of redundantly actuated parallel manipulators," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 1153–1158, IEEE, 2006.

[80] M. Mahvash and P. E. Dupont, "Stiffness control of surgical continuum manipulators," *Robotics, IEEE Transactions on*, vol. 27, no. 2, pp. 334–345, 2011.

[81] N. Hogan, "Impedance control: An approach to manipulation," in *American Control Conference, 1984*, pp. 304–313, IEEE, 1984.

[82] N. Hogan, "Impedance control of industrial robots," *Robotics and Computer-Integrated Manufacturing*, vol. 1, no. 1, pp. 97–113, 1984.

[83] N. Hogan, "Impedance control: An approach to manipulation: Part ii—implementation," *Journal of dynamic systems, measurement, and control*, vol. 107, no. 1, pp. 8–16, 1985.

[84] N. Hogan, "Stable execution of contact tasks using impedance control," in *Robotics and Automation. Proceedings. 1987 IEEE International Conference on*, vol. 4, pp. 1047–1054, IEEE, 1987.

[85] S. Tachi, T. Sakaki, H. Arai, S. Nishizawa, and J. F. Pelaez-Polo, "Impedance control of a direct-drive manipulator without using force sensors," *Advanced Robotics*, vol. 5, no. 2, pp. 183–205, 1990.

[86] W.-S. Lu and Q.-H. Meng, "Impedance control with adaptation for robotic manipulations," *Robotics and Automation, IEEE Transactions on*, vol. 7, no. 3, pp. 408–415, 1991.

[87] T. Lasky and T. Hsia, "On force-tracking impedance control of robot manipulators," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pp. 274–280, IEEE, 1991.

[88] S. Lee and H. S. Lee, "Intelligent control of manipulators interacting with an uncertain environment based on generalized impedance," in *Intelligent Control, 1991., Proceedings of the 1991 IEEE International Symposium on*, pp. 61–66, IEEE, 1991.

[89] R. Colbaugh, H. Seraji, and K. Glass, "Impedance control for dexterous space manipulators," in *Decision and Control, 1992., Proceedings of the 31st IEEE Conference on*, pp. 1881–1886, IEEE, 1992.

[90] S. A. Schneider and R. H. Cannon Jr, "Object impedance control for cooperative manipulation: Theory and experimental results," *Robotics and Automation, IEEE Transactions on*, vol. 8, no. 3, pp. 383–394, 1992.

[91] A. DeHon, T. Knight Jr, and T. Simon, "Automatic impedance control," in *Solid-State Circuits Conference, 1993. Digest of Technical Papers. 40th ISSCC., 1993 IEEE International*, pp. 164–165, IEEE, 1993.

[92] R. Colbaugh, H. Seraji, and K. Glass, "Direct adaptive impedance control of robot manipulators," *Journal of Robotic Systems*, vol. 10, no. 2, pp. 217–248, 1993.

[93] T. Murakami, R. Nakamura, F. Yu, and K. Ohnishi, "Force sensorless impedance control by disturbance observer," in *Power Conversion Conference, 1993. Yokohama 1993., Conference Record of the*, pp. 352–357, IEEE, 1993.

[94] R. Ikeura and H. Inooka, "Variable impedance control of a robot for cooperation with a human," in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 3, pp. 3097–3102, IEEE, 1995.

[95] R. G. Bonitz and T. C. Hsia, "Internal force-based impedance control for cooperating manipulators," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 1, pp. 78–89, 1996.

[96] H. Seraji and R. Colbaugh, "Force tracking in impedance control," *The International Journal of Robotics Research*, vol. 16, no. 1, pp. 97–117, 1997.

[97] C.-C. Cheah and D. Wang, "Learning impedance control for robotic manipulators," *Robotics and Automation, IEEE Transactions on*, vol. 14, no. 3, pp. 452–465, 1998.

[98] G. Morel, E. Malis, and S. Boudet, "Impedance based combination of visual and force control," in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 2, pp. 1743–1748, IEEE, 1998.

[99] A. Albu-Schaffer and G. Hirzinger, "Cartesian impedance control techniques for torque controlled light-weight robots," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 1, pp. 657–663, IEEE, 2002.

[100] R. Ikeura, T. Moriguchi, and K. Mizutani, "Optimal variable impedance control for a robot and its application to lifting an object with a human," in *Robot and Human Interactive Communication, 2002. Proceedings. 11th IEEE International Workshop on*, pp. 500–505, IEEE, 2002.

[101] S. Jung, T. C. Hsia, and R. G. Bonitz, "Force tracking impedance control of robot manipulators under unknown environment," *Control Systems Technology, IEEE Transactions on*, vol. 12, no. 3, pp. 474–483, 2004.

[102] Y. Zhu and E. J. Barth, "Impedance control of a pneumatic actuator for contact tasks," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 987–992, IEEE, 2005.

[103] G. Vossoughi and A. Karimzadeh, "Impedance control of a two degree-of-freedom planar flexible link manipulator using singular perturbation theory," *Robotica*, vol. 24, no. 02, pp. 221–228, 2006.

[104] A. Albu-Schäffer, C. Ott, and G. Hirzinger, "A unified passivity-based control framework for position, torque and impedance control of flexible joint robots," *The International Journal of Robotics Research*, vol. 26, no. 1, pp. 23–39, 2007.

[105] K. Lee and M. Buss, "Force tracking impedance control with variable target stiffness," in *Proceedings of the 17th IFAC World Congress*, vol. 17, pp. 6751–6756, 2008.

[106] H. Jianbin, X. Zongwu, J. Minghe, J. Zainan, and L. Hong, "Adaptive impedance-controlled manipulator based on collision detection," *Chinese Journal of Aeronautics*, vol. 22, no. 1, pp. 105–112, 2009.

[107] T. Wongratanaphisan and M. Cole, "Robust impedance control of a flexible structure mounted manipulator performing contact tasks," *Robotics, IEEE Transactions on*, vol. 25, no. 2, pp. 445–451, 2009.

[108] H. Mehdi and O. Boubaker, "Stiffness and impedance control using lyapunov theory for robot-aided rehabilitation," *International Journal of Social Robotics*, vol. 4, no. 1, pp. 107–119, 2012.

[109] R. Carelli and J. Postigo, "An adaptive impedance controller for robot manipulators," *Algorithms and Architectures for Real-Time Control 1991*, p. 87, 2014.

[110] H. Seraji, "Adaptive admittance control: An approach to explicit force control in compliant motion," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pp. 2705–2712, IEEE, 1994.

[111] W. S. Newman and Y. Zhang, "Stable interaction control and coulomb friction compensation using natural admittance control," *Journal of robotic systems*, vol. 11, no. 1, pp. 3–11, 1994.

[112] G. D. Glosser and W. S. Newman, "The implementation of a natural admittance controller on an industrial manipulator," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pp. 1209–1215, IEEE, 1994.

[113] R. Van der Linde and P. Lammertse, "Hapticmaster-a generic force controlled robot for human interaction," *Industrial Robot: An International Journal*, vol. 30, no. 6, pp. 515–524, 2003.

[114] F. Augugliaro and R. D'Andrea, "Admittance control for physical human-quadrocopter interaction," in *Control Conference (ECC), 2013 European*, pp. 1805–1810, IEEE, 2013.

[115] C. Ott, R. Mukherjee, and Y. Nakamura, "A hybrid system framework for unified impedance and admittance control," *Journal of Intelligent & Robotic Systems*, pp. 1–17, 2014.

[116] D. E. Whitney, "Force feedback control of manipulator fine motions," *Journal of Dynamic Systems, Measurement, and Control*, vol. 99, no. 2, pp. 91–97, 1977.

[117] R. Volpe and P. Khosla, "An experimental evaluation and comparison of explicit force control strategies for robotic manipulators," in *American Control Conference, 1992*, pp. 758–765, IEEE, 1992.

[118] D. Dawson, F. Lewis, and J. Dorsey, "Robust force control of a robot manipulator," *The International journal of robotics research*, vol. 11, no. 4, pp. 312–319, 1992.

[119] R. Volpe and P. Khosla, "A theoretical and experimental investigation of explicit force control strategies for manipulators," *Automatic Control, IEEE Transactions on*, vol. 38, no. 11, pp. 1634–1650, 1993.

[120] S. Jung and T. Hsia, "Robust neural force control scheme under uncertainties in robot dynamics and unknown environment," *Industrial Electronics, IEEE Transactions on*, vol. 47, no. 2, pp. 403–412, 2000.

[121] J. Kövecses, L. L. Kovács, and G. Stépán, "Dynamics modeling and stability of robotic systems with discrete-time force control," *Archive of Applied Mechanics*, vol. 77, no. 5, pp. 293–299, 2007.

[122] N. Zemiti, G. Morel, T. Ortmaier, and N. Bonnet, "Mechatronic design of a new robot for force control in minimally invasive surgery," *Mechatronics, IEEE/ASME Transactions on*, vol. 12, no. 2, pp. 143–153, 2007.

[123] T. Murakami, F. Yu, and K. Ohnishi, "A dynamical approach to force control without force sensor," *Robotics, Mechatronics and Manufacturing Systems*, p. 333, 2012.

[124] W. Xu, C. Cai, and Y. Zou, "Neural-network-based robot time-varying force control with uncertain manipulator–environment system," *Transactions of the Institute of Measurement and Control*, vol. 36, no. 8, pp. 999–1009, 2014.

[125] C. Shin, S. Moon, J. Kwon, J. Huh, and D. Hong, "Force control of cleaning tool system for building wall maintenance robot on built-in guide rail," in *Proc. of the 31st International Symposium on Automation and Robotics in Construction*, pp. 157–162, 2014.

[126] J. J. Craig and M. H. Raibert, "A systematic method of hybrid position/force control of a manipulator," in *Computer Software and Applications Conference, 1979. Proceedings. COMPSAC 79. The IEEE Computer Society's Third International*, pp. 446–451, 1979.

[127] M. H. Raibert and J. J. Craig, "Hybrid position/force control of manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 103, no. 2, pp. 126–133, 1981.

[128] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 1, pp. 43–53, 1987.

[129] H. Faessler, "Manipulators constrained by stiff contact: dynamics, control, and experiments," *The International Journal of Robotics Research*, vol. 9, no. 4, pp. 40–58, 1990.

[130] J. T. Wen and K. Kreutz-Delgado, "Motion and force control of multiple robotic manipulators," *Automatica*, vol. 28, no. 4, pp. 729–743, 1992.

[131] Z. Lu and A. A. Goldenberg, "Implementation of robust impedance and force control," *Journal of Intelligent and Robotic Systems*, vol. 6, no. 2-3, pp. 145–163, 1992.

[132] S. Chiaverini, B. Siciliano, and L. Villani, "A stable force/position controller for robot manipulators," in *Decision and Control, 1992., Proceedings of the 31st IEEE Conference on*, pp. 1869–1874, IEEE, 1992.

[133] S. Chiaverini and L. Sciavicco, "The parallel approach to force/position control of robotic manipulators," *Robotics and Automation, IEEE Transactions on*, vol. 9, no. 4, pp. 361–373, 1993.

[134] D. Wang and N. H. McClamroch, "Position and force control for constrained manipulator motion: Lyapunov's direct method," *Robotics and Automation, IEEE Transactions on*, vol. 9, no. 3, pp. 308–313, 1993.

[135] S. Chiaverini, B. Siciliano, and L. Villani, "Force/position regulation of compliant robot manipulators," *Automatic Control, IEEE Transactions on*, vol. 39, no. 3, pp. 647–652, 1994.

[136] R. M. DeSantis, "Motion/force control of robotic manipulators," *Journal of dynamic systems, measurement, and control*, vol. 118, no. 2, pp. 386–389, 1996.

[137] M. K. Vukobratović and Y. Ekalo, "New approach to control of robotic manipulators interacting with dynamic environment," *Robotica*, vol. 14, no. 01, pp. 31–39, 1996.

[138] K. Kiguchi and T. Fukuda, "Intelligent position/force controller for industrial robot manipulators: Application of fuzzy neural networks: Neural networks for robot control," *IEEE Transactions on Industrial Electronics*, vol. 44, no. 6, pp. 753–761, 1997.

[139] B. Siciliano and L. Villani, "An output feedback parallel force/position regulator for a robot manipulator in contact with a compliant environment," *Systems & control letters*, vol. 29, no. 5, pp. 295–300, 1997.

[140] K. S. Eom, I. H. Suh, W. K. Chung, and S.-R. Oh, "Disturbance observer based force control of robot manipulator without force sensor," in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 4, pp. 3012–3017, IEEE, 1998.

[141] S. Chiaverini, B. Siciliano, and L. Villani, "Force and position tracking: Parallel control with stiffness adaptation," *Control Systems, IEEE*, vol. 18, no. 1, pp. 27–33, 1998.

[142] I.-C. Lin and L.-C. Fu, "Adaptive hybrid force/position control of a flexible manipulator for automated deburring with online cutting trajectory modification," in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 1, pp. 818–825, IEEE, 1998.

[143] L. Villani, C. Canudas de Wit, and B. Brogliato, "An exponentially stable adaptive control for force and position tracking of robot manipulators," *Automatic Control, IEEE Transactions on*, vol. 44, no. 4, pp. 798–802, 1999.

[144] D. Xiao, B. K. Ghosh, N. Xi, and T. J. Tarn, "Sensor-based hybrid position/force control of a robot manipulator in an uncalibrated environment," *Control Systems Technology, IEEE Transactions on*, vol. 8, no. 4, pp. 635–645, 2000.

[145] K. Kiguchi and T. Fukuda, "Position/force control of robot manipulators for geometrically unknown objects using fuzzy neural networks," *Industrial Electronics, IEEE Transactions on*, vol. 47, no. 3, pp. 641–649, 2000.

[146] L. Villani, C. Natale, B. Siciliano, and C. Canudas de Wit, "An experimental study of adaptive force/position control algorithms for an industrial robot," *Control Systems Technology, IEEE Transactions on*, vol. 8, no. 5, pp. 777–786, 2000.

[147] C.-C. Cheah, S. Kawamura, and S. Arimoto, "Stability of hybrid position and force control for robotic manipulator with kinematics and dynamics uncertainties," *Automatica*, vol. 39, no. 5, pp. 847–855, 2003.

[148] E.-c. LI and W. LI, "Hybrid force/position control for positional-controlled robotic manipulators in unknown environment [j]," *Journal of China Coal Society*, vol. 6, p. 023, 2007.

[149] Y. Karayiannidis, G. Rovithakis, and Z. Doulgeri, "Force/position tracking for a robotic manipulator in compliant contact with a surface using neuro-adaptive control," *Automatica*, vol. 43, no. 7, pp. 1281–1288, 2007.

[150] H. Shao-Hui, C. Xiao-Tao, and L. Yuan-Chun, "Force/position control algorithm for manipulator based on wavelet network [j]," *Journal of Jilin University (Engineering and Technology Edition)*, vol. 1, p. 033, 2008.

[151] A. Zuev and V. Filaretov, "Features of designing combined force/position manipulator control systems," *Journal of Computer and Systems Sciences International*, vol. 48, no. 1, pp. 146–154, 2009.

[152] D. Nganga-Kouya, M. Saad, and F. A. Okou, "A novel adaptive hybrid force-position control of a robotic manipulator," *International Journal of Modelling, Identification and Control*, vol. 13, no. 1, pp. 97–107, 2011.

[153] N. Kumar, V. Panwar, N. Sukavanam, S. P. Sharma, and J.-H. Borm, "Neural network based hybrid force/position control for robot manipulators," *International Journal of Precision Engineering and Manufacturing*, vol. 12, no. 3, pp. 419–426, 2011.

[154] C. P. Bechlioulis, Z. Doulgeri, and G. A. Rovithakis, "Guaranteeing prescribed performance and contact maintenance via an approximation free robot force/position controller," *Automatica*, vol. 48, no. 2, pp. 360–365, 2012.

[155] P. Gierlak, "Hybrid position/force control of the scorbot-er 4pc manipulator with neural compensation of nonlinearities," in *Artificial Intelligence and Soft Computing*, pp. 433–441, Springer, 2012.

[156] B. Achili, B. Daachi, Y. Amirat, A. Ali-Cherif, and M. Daachi, "A stable adaptive force/position controller for a c5 parallel robot: a neural network approach," *Robotica*, vol. 30, no. 07, pp. 1177–1187, 2012.

[157] H. Chaudhary, V. Panwar, R. Prasad, and N. Sukavanam, "Adaptive neuro fuzzy based hybrid force/position control for an industrial robot manipulator," *Journal of Intelligent Manufacturing*, pp. 1–10, 2014.

[158] L. Zhang, Q. X. Jia, G. Chen, and H. X. Sun, "The analysis of free-floating space manipulator when tracking desired force/position," in *Applied Mechanics and Materials*, vol. 742, pp. 560–566, Trans Tech Publ, 2015.

[159] S. A. M. Dehghan, M. Danesh, and F. Sheikholeslam, "Adaptive hybrid force/position control of robot manipulators using an adaptive force estimator in the presence of parametric uncertainty," *Advanced Robotics*, vol. 29, no. 4, pp. 209–223, 2015.

[160] M. Al Mashagbeh and M. B. Khamesee, "Force control for position interface industrial manipulator working in unknown environment," *Microsystem Technologies*, pp. 1–7, 2015.

[161] J. Cazalilla, M. Vallés, A. Valera, V. Mata, and M. Díaz-Rodríguez, "Implementation of force and position controllers for a 3dof parallel manipulator," in *Multibody Mechatronic Systems*, pp. 359–369, Springer, 2015.

[162] D. Heck, A. Saccon, N. van de Wouw, and H. Nijmeijer, "Switching control for tracking of a hybrid position-force trajectory," *arXiv preprint arXiv:1503.00603*, 2015.

[163] C. H. An and J. M. Hollerbach, "The role of dynamic models in cartesian force control of manipulators," *The International Journal of Robotics Research*, vol. 8, no. 4, pp. 51–72, 1989.

[164] O. Khatib and J. Burdick, "Motion and force control of robot manipulators," in *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, vol. 3, pp. 1381–1386, IEEE, 1986.

[165] T. Yoshikawa, "Dynamic hybrid position/force control of robot manipulators–description of hand constraints and calculation of joint driving force," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 5, pp. 386–392, 1987.

[166] T. Yoshikawa, T. Sugie, and M. Tanaka, "Dynamic hybrid position/force control of robot manipulators-controller design and experiment," *Robotics and Automation, IEEE Journal of*, vol. 4, no. 6, pp. 699–705, 1988.

[167] N. H. McClamroch and D. Wang, "Feedback stabilization and tracking of constrained robots," in *American Control Conference, 1987*, pp. 464–469, IEEE, 1987.

[168] N. H. McClamroch and D. Wang, "Feedback stabilization and tracking of constrained robots," *Automatic Control, IEEE Transactions on*, vol. 33, no. 5, pp. 419–426, 1988.

[169] R. K. Kankaanranta and H. N. Koivo, "Dynamics and simulation of compliant motion of a manipulator," *Robotics and Automation, IEEE Journal of*, vol. 4, no. 2, pp. 163–173, 1988.

[170] G. Fodor and G. Tevesz, "Hybrid position and force control algorithm expansion of a robot control system," *Electrical Engineering*, vol. 43, no. 4, pp. 251–261, 1999.

[171] Y. Sun, A. Behal, and C.-K. R. Chung, *New Development in Robot Vision*, vol. 23. Springer, 2014.

[172] L. Sciavicco and B. Siciliano, *Modelling and control of robot manipulators*. Springer Science & Business Media, 2012.

[173] S. Chiaverini, B. Siciliano, and L. Villani, "A survey of robot interaction control schemes with experimental comparison," *IEEE/ASME Transactions on mechatronics*, vol. 4, no. 3, pp. 273–285, 1999.

[174] F. Cisneros, "Engineering manager," *Yaskawa-Motoman Mexico. Personal communication*, 2003.

[175] E. M. Mouaddib, R. Sagawa, T. Echigo, and Y. Yagi, "Stereovision with a single camera and multiple mirrors," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pp. 800–805, IEEE, 2005.

[176] E. J. González-Galván, S. R. Cruz-Ramırez, M. J. Seelinger, and J. J. Cervantes-Sánchez, "An efficient multi-camera, multi-target scheme for the three-dimensional control of robots using uncalibrated vision," *Robotics and Computer-Integrated Manufacturing*, vol. 19, no. 5, pp. 387–400, 2003.

[177] J. Davis and X. Chen, "Calibrating pan-tilt cameras in widearea surveillance networks," in *In IEEE International Conference on Computer Vision*, Citeseer, 2003.

[178] T. Thormählen and H. Broszio, "Automatic line-based estimation of radial lens distortion," *Integrated Computer-Aided Engineering*, vol. 12, no. 2, pp. 177–190, 2005.

[179] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[180] C. Shu, A. Brunton, and M. Fiala, *Automatic grid finding in calibration patterns using Delaunay triangulation.* National Research Council of Canada, 2003.

[181] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision.* Cambridge university press, 2003.

[182] A. Fusiello, E. Trucco, and A. Verri, "A compact algorithm for rectification of stereo pairs," *Machine Vision and Applications*, vol. 12, no. 1, pp. 16–22, 2000.

[183] A. Burton and J. Radford, *Thinking in perspective: critical essays in the study of thought processes*. Methuen, 1978.

[184] D. H. Warren and E. R. Strelow, *Electronic Spatial Sensing for the Blind: Contributions from Perception, Rehabilitation, and Computer Vision*, vol. 99. Springer Science & Business Media, 2013.

[185] J. J. Gibson, "The perception of the visual world.," 1950.

[186] C. S. Royden and K. D. Moore, "Use of speed cues in the detection of moving objects by moving observers," *Vision research*, vol. 59, pp. 17–24, 2012.

[187] K. R. Aires, A. M. Santana, and A. A. Medeiros, "Optical flow using color information: preliminary results," in *Proceedings of the 2008 ACM symposium on Applied computing*, pp. 1607–1611, ACM, 2008.

[188] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM computing surveys (CSUR)*, vol. 27, no. 3, pp. 433–466, 1995.

[189] D. Fleet and Y. Weiss, "Optical flow estimation," in *Handbook of mathematical models in computer vision*, pp. 237–257, Springer, 2006.

[190] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *International journal of computer vision*, vol. 12, no. 1, pp. 43–77, 1994.

[191] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, 2011.

[192] G. L. Barrows, J. S. Chahl, and M. V. Srinivasan, "Biologically inspired visual sensing and flight control," *The Aeronautical Journal*, vol. 107, no. 1069, pp. 159–168, 2003.

[193] B. D. Lucas, T. Kanade, *et al.*, "An iterative image registration technique with an application to stereo vision," 1981.

[194] B. D. Lucas, "Generalized image matching by the method of differences.," 1986.

[195] J. Shi *et al.*, "Good features to track," in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pp. 593–600, IEEE, 1994.

[196] J.-Y. Bouguet, "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm," *Intel Corporation*, vol. 5, no. 1-10, p. 4, 2001.

[197] V. Zanotto, A. Gasparetto, A. Lanzutti, P. Boscariol, and R. Vidoni, "Experimental validation of minimum time-jerk algorithms for industrial robots," *Journal of Intelligent & Robotic Systems*, vol. 64, no. 2, pp. 197–219, 2011.

[198] S. Jeon, M. Tomizuka, and T. Katou, "Kinematic kalman filter (kkf) for robot end-effector sensing," *Journal of dynamic systems, measurement, and control*, vol. 131, no. 2, p. 021010, 2009.

[199] R. Merry, M. Van de Molengraft, and M. Steinbuch, "Velocity and acceleration estimation for optical incremental encoders," *Mechatronics*, vol. 20, no. 1, pp. 20–26, 2010.

[200] C. Dumas, S. Caro, S. Garnier, and B. Furet, "Joint stiffness identification of six-revolute industrial serial robots," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 4, pp. 881–888, 2011.

[201] H. Tanaka, H. Nishi, and K. Ohnishi, "An approach to acceleration estimation using fpga," in *Industrial Electronics, 2008. ISIE 2008. IEEE International Symposium on*, pp. 1959–1964, IEEE, 2008.

[202] W.-H. Zhu and T. Lamarche, "Velocity estimation by using position and acceleration sensors," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 5, pp. 2706–2715, 2007.

[203] K. Kozłowski and P. Herman, "Control of robot manipulators in terms of quasi-velocities," *Journal of Intelligent & Robotic Systems*, vol. 53, no. 3, pp. 205–221, 2008.

[204] A. Nikoobin and R. Haghighi, "Lyapunov-based nonlinear disturbance observer for serial n-link robot manipulators," *Journal of Intelligent & Robotic Systems*, vol. 55, no. 2, pp. 135–153, 2009.

[205] M. Moradi and H. Malekizade, "Neural network identification based multivariable feedback linearization robust control for a two-link manipulator," *Journal of Intelligent & Robotic Systems*, vol. 72, no. 2, p. 167, 2013.

[206] M. F. Lima, J. T. Machado, and M. Crisóstomo, "Filtering method in backlash phenomena analysis," *Mathematical and Computer Modelling*, vol. 49, no. 7, pp. 1494–1503, 2009.

[207] C. Rodriguez-Donate, L. Morales-Velazquez, R. A. Osornio-Rios, G. Herrera-Ruiz, and R. d. J. Romero-Troncoso, "Fpga-based fused smart sensor for dynamic and vibration parameter extraction in industrial robot links," *Sensors*, vol. 10, no. 4, pp. 4114–4129, 2010.

[208] L. Morales-Velazquez, R. de Jesus Romero-Troncoso, R. A. Osornio-Rios, and E. Cabal-Yepez, "Sensorless jerk monitoring using an adaptive antisymmetric high-order fir filter," *Mechanical Systems and Signal Processing*, vol. 23, no. 7, pp. 2383–2394, 2009.

[209] G. G. Rigatos, "Particle filtering for state estimation in nonlinear industrial systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 11, pp. 3885–3900, 2009.

[210] P. Cheng and B. Oelmann, "Joint-angle measurement using accelerometers and gyroscopes—a survey," *IEEE Transactions on instrumentation and measurement*, vol. 59, no. 2, pp. 404–414, 2010.

[211] C. Rodriguez-Donate, R. A. Osornio-Rios, J. R. Rivera-Guillen, and R. d. J. Romero-Troncoso, "Fused smart sensor network for multi-axis forward kinematics estimation in industrial robots," *Sensors*, vol. 11, no. 4, pp. 4335–4357, 2011.

[212] H. Musoff and P. Zarchan, *Fundamentals of Kalman Filtering: A practical approach*. American Institute of Aeronautics and Astronautics, 2005.

[213] D. M. Wolpert and Z. Ghahramani, "Computational principles of movement neuroscience.," *Nature neuroscience*, vol. 3, no. 11, 2000.

[214] S. Drakunov, "An adaptive quasioptimal filter with discontinuous parameters," *Avtomatika i Telemekhanika*, no. 9, pp. 76–86, 1983.

[215] J. J. Craig, *Introduction to robotics: mechanics and control*, vol. 3. Pearson Prentice Hall Upper Saddle River, 2005.

[216] S. K. Dwivedy and P. Eberhard, "Dynamic analysis of flexible manipulators, a literature review," *Mechanism and machine theory*, vol. 41, no. 7, pp. 749–777, 2006.

[217] V. Jain, *Micromanufacturing Processes*. CRC Press, 2016.

[218] T. Yamaguchi, M. Hirata, and J. C. K. Pang, *High-speed precision motion control*. CRC press, 2017.

[219] J. Brunetti, F. Massi, A. Saulot, M. Renouf, and W. DAmbrogio, "System dynamic instabilities induced by sliding contact: A numerical analysis with experimental validation," *Mechanical Systems and Signal Processing*, vol. 58, pp. 70 – 86, 2015.

[220] A. Cowley and A. Boyle, "Active dampers for machine tools," *Annals of the CIRP*, vol. 18, no. 1, pp. 213–222, 1970.

[221] D. Karnopp, "Active damping in road vehicle suspension systems," *Vehicle System Dynamics*, vol. 12, no. 6, pp. 291–311, 1983.

[222] B. Yan, M. Brennan, S. Elliott, and N. Ferguson, "Active vibration isolation of a system with a distributed parameter isolator using absolute velocity feedback control," *Journal of Sound and Vibration*, vol. 329, no. 10, pp. 1601 – 1614, 2010.

[223] M. Zaeh, R. Kleinwort, P. Fagerer, and Y. Altintas, "Automatic tuning of active vibration control systems using inertial actuators," *CIRP Annals-Manufacturing Technology*, 2017.

[224] A. Tiwari, M. Lathkar, P. Shendge, and S. Phadke, "Skyhook control for active suspension system of heavy duty vehicles using inertial delay control," in *Power Electronics, Intelligent Control and Energy Systems (ICPEICES), IEEE International Conference on*, pp. 1–6, IEEE, 2016.

[225] S. Song, F. Jiang, H. Shi, and Y. Peng, "Skyhook-based body-seat system preview control for a semi-active suspension tractor," *Sensors & Transducers*, vol. 179, no. 9, p. 60, 2014.

[226] M. Gavagni, P. Gardonio, and S. J. Elliot, "Design and fabrication of a micro-velocity sensor for direct velocity feedback control systems," in *Proc. 6th Int. Symp. Active Noise Vibrat. Control*, pp. 1–16, 2006.

[227] Y. Zheng, J. Liu, X. Liu, D. Fang, and L. Wu, "Adaptive second-order sliding mode control design for a class of nonlinear systems with unknown input," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[228] F. Beltran-Carbajal, A. Valderrabano-Gonzalez, J. C. Rosas-Caro, and A. Favela-Contreras, "An asymptotic differentiation approach of signals in velocity tracking control of dc motors," *Electric Power Systems Research*, vol. 122, pp. 218–223, 2015.

[229] A. Levant, "Higher-order sliding modes, differentiation and output-feedback control," *International journal of Control*, vol. 76, no. 9-10, pp. 924–941, 2003.

[230] A. F. de Loza, J. Cieslak, D. Henry, J. Dávila, and A. Zolghadri, "Sensor fault diagnosis using a non-homogeneous high-order sliding mode observer

with application to a transport aircraft," *IET Control Theory & Applications*, vol. 9, no. 4, pp. 598–607, 2015.

[231] R. Pandey, "Design of wavelet-based differentiator filter," *Journal of International Academy Of Physical Sciences*, vol. 17, no. 1, 2013.

[232] E. K. Olama and S. Valiloo, "A fast wavelet denoising method," in *2011 3rd International Conference on Computer Research and Development*, vol. 1, pp. 492–494, March 2011.

[233] T. E. Duncan, P. Mandl, and B. Pasik-Duncan, "Numerical differentiation and parameter estimation in higher-order linear stochastic systems," *IEEE Transactions on Automatic control*, vol. 41, no. 4, pp. 522–532, 1996.

[234] S. Ibrir and S. Diop, "A numerical procedure for filtering and efficient high-order signal differentiation," *International Journal of Applied Mathematics and Computer Science*, vol. 14, no. 2, pp. 201–208, 2004.

[235] F. Esfandiari and H. K. Khalil, "Output feedback stabilization of fully linearizable systems," *International Journal of control*, vol. 56, no. 5, pp. 1007–1037, 1992.

[236] N. Boizot, E. Busvelle, and J. Sachau, "High-gain observers and kalman filtering in hard real-time," in *RTL 9th Workshop*, 2007.

[237] Y. Chen and B. M. Vinagre, "A new iir-type digital fractional order differentiator," *Signal Processing*, vol. 83, no. 11, pp. 2359–2365, 2003.

[238] W.-D. Chang and D.-M. Chang, "Design of a higher-order digital differentiator using a particle swarm optimization approach," *Mechanical Systems and Signal Processing*, vol. 22, no. 1, pp. 233–247, 2008.

[239] C.-K. Chen and J.-H. Lee, "Design of high-order digital differentiators using l/sub 1/error criteria," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 42, no. 4, pp. 287–291, 1995.

[240] A. Sabatini, "Real-time kalman filter applied to biomechanical data for state estimation and numerical differentiation," *Medical and Biological Engineering and Computing*, vol. 41, no. 1, pp. 2–10, 2003.

[241] M. Sharifi, M. Seif, and M. Hadi, "A comparison between numerical differentiation and kalman filtering for a leo satellite velocity determination," *Artificial Satellites*, vol. 48, no. 3, pp. 103–110, 2013.

[242] J. Moore, "Optimum differentiation using kalman filter theory," *Proceedings of the IEEE*, vol. 56, no. 5, pp. 871–871, 1968.

[243] V. I. Utkin, *Sliding modes in control and optimization*. Springer Science & Business Media, 2013.

[244] A. Levant, "Robust exact differentiation via sliding mode technique," *automatica*, vol. 34, no. 3, pp. 379–384, 1998.

[245] W. T. Higgins, "A comparison of complementary and kalman filtering," *IEEE Transactions on Aerospace and Electronic Systems*, no. 3, pp. 321–325, 1975.

[246] L. J. Puglisi, "On the velocity and acceleration estimation from discrete time-position signal of linear encoders," *Journal of Control Engineering and Applied Informatics*, vol. 17, no. 3, pp. 30–40, 2015.

[247] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," *IEEE transactions on information theory*, vol. 36, no. 5, pp. 961–1005, 1990.

[248] M. N. Do and M. Vetterli, "Texture similarity measurement using kullback-leibler distance on wavelet subbands," in *Image Processing, 2000. Proceedings. 2000 International Conference on*, vol. 3, pp. 730–733, IEEE, 2000.

[249] D. Heric and D. Zazula, "Combined edge detection using wavelet transform and signal registration," *Image and Vision computing*, vol. 25, no. 5, pp. 652–662, 2007.

[250] R. Yan, R. X. Gao, and X. Chen, "Wavelets for fault diagnosis of rotary machines: A review with applications," *Signal processing*, vol. 96, pp. 1–15, 2014.

[251] T. Abuhamdia, S. Taheri, and J. Burns, "Laplace wavelet transform theory and applications," *Journal of Vibration and Control*, vol. 0, no. 0, p. 1077546317707103, 2017.

[252] L. Ebadi, Z. Helmi, M. Shafri, S. B. Mansor, and R. Ashurov, "A review of applying second-generation wavelets for noise removal from remote sensing data," *Environmental earth sciences*, vol. 70, no. 6, p. 2679, 2013.

[253] M. A. Kabir and C. Shahnaz, "Denoising of ecg signals based on noise reduction algorithms in emd and wavelet domains," *Biomedical Signal Processing and Control*, vol. 7, no. 5, pp. 481–489, 2012.

[254] L. R. Watkins, "Review of fringe pattern phase recovery using the 1-d and 2-d continuous wavelet transforms," *Optics and Lasers in Engineering*, vol. 50, no. 8, pp. 1015–1022, 2012.

[255] T. Abuhamdia and S. Taheri, "Wavelets as a tool for systems analysis and control," *Journal of Vibration and Control*, vol. 23, no. 9, pp. 1377–1416, 2017.

[256] J.-M. Combes, A. Grossmann, and P. Tchamitchian, *Wavelets: Time-Frequency Methods and Phase Space Proceedings of the International Conference, Marseille, France, December 14–18, 1987*. Springer Science & Business Media, 2012.

[257] C. K. Chui and G. Chen, "Wavelet kalman filtering," in *Kalman Filtering*, pp. 171–183, Springer, 2017.

[258] X. Chen, C. Shen, W.-b. Zhang, M. Tomizuka, Y. Xu, and K. Chiu, "Novel hybrid of strong tracking kalman filter and wavelet neural network for gps/ins during gps outages," *Measurement*, vol. 46, no. 10, pp. 3847–3854, 2013.

[259] P. Sun, X. Tang, Y. Huang, and G. Sun, "Wavelet de-noising kalman filter-based global navigation satellite system carrier tracking in the presence of ionospheric scintillation," *IET Radar, Sonar & Navigation*, vol. 11, no. 2, pp. 226–234, 2016.

[260] Y. Chen, Q. Zhao, B. Hu, J. Li, H. Jiang, W. Lin, Y. Li, S. Zhou, and H. Peng, "A method of removing ocular artifacts from eeg using discrete wavelet transform and kalman filtering," in *Bioinformatics and Biomedicine (BIBM), 2016 IEEE International Conference on*, pp. 1485–1492, IEEE, 2016.

[261] S. Butterworth, "Experimental wireless and the wireless engineer," *Wireless Eng*, vol. 7, p. 536, 1930.

[262] P. Gaydecki, *Foundations of digital signal processing: theory, algorithms and hardware design*, vol. 15. Iet, 2004.

[263] B. Yu, D. Gabriel, L. Noble, and K.-N. An, "Estimate of the optimum cut-off frequency for the butterworth low-pass digital filter," *Journal of Applied Biomechanics*, vol. 15, no. 3, pp. 318–329, 1999.

[264] J. H. Challis, "A procedure for the automatic determination of filter cut-off frequency for the processing of biomechanical data," *Journal of Applied Biomechanics*, vol. 15, no. 3, pp. 303–317, 1999.

[265] P. Zarchan and H. Musoff, *Fundamentals of Kalman Filtering: A Practical Approach*. No. v. 190 in Fundamentals of Kalman filtering: a practical approach, American Institute of Aeronautics and Astronautics, Incorporated, 2000.

[266] M. Euston, P. Coote, R. Mahony, J. Kim, and T. Hamel, "A complementary filter for attitude estimation of a fixed-wing uav," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pp. 340–345, IEEE, 2008.

[267] Y. Tian, H. Wei, and J. Tan, "An adaptive-gain complementary filter for real-time human motion tracking with marg sensors in free-living environments,"

IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 21, no. 2, pp. 254–264, 2013.

[268] R. Kottath, P. Narkhede, V. Kumar, V. Karar, and S. Poddar, "Multiple model adaptive complementary filter for attitude estimation," *Aerospace Science and Technology (2017)*, 2017.

[269] E. Chalmers, J. Le, D. Sukhdeep, J. Watt, J. Andersen, and E. Lou, "Inertial sensing algorithms for long-term foot angle monitoring for assessment of idiopathic toe-walking," *Gait & Posture*, vol. 39, no. 1, pp. 485 – 489, 2014.

[270] S. Winder, "Chapter 9 - phase-shift networks (all-pass filters)," in *Analog and Digital Filter Design (Second Edition)* (S. Winder, ed.), EDN Series for Design Engineers, pp. 243 – 284, Woburn: Newnes, second edition ed., 2002.

[271] A. Bariska, "Time machine, anyone?," Mar. 2008.

[272] A. K. Aniyan, N. S. Philip, V. J. Samar, J. A. Desjardins, and S. J. Segalowitz, "A wavelet based algorithm for the identification of oscillatory event-related potential components," *Journal of Neuroscience Methods*, vol. 233, no. Supplement C, pp. 63 – 72, 2014.

[273] I. Daubechies, "Ten lectures on wavelets, vol. 61 of cbms-nsf regional conference series in applied mathematics," 1992.

[274] R. Gade and T. B. Moeslund, "Thermal tracking of sports players," *Sensors*, vol. 14, no. 8, pp. 13679–13691, 2014.

[275] X. Li, R. Guo, and C. Chen, "Robust pedestrian tracking and recognition from flir video: A unified approach via sparse coding," *Sensors*, vol. 14, no. 6, pp. 11245–11259, 2014.

[276] C. S. Royden and E. M. Connors, "The detection of moving objects by moving observers," *Vision research*, vol. 50, no. 11, pp. 1014–1024, 2010.

[277] C. S. Royden and M. A. Holloway, "Detecting moving objects in an optic flow field using direction-and speed-tuned operators," *Vision research*, vol. 98, pp. 14–25, 2014.

[278] S. Shantaiya, K. Verma, and K. Mehta, "Multiple object tracking using kalman filter and optical flow," *European Journal of Advances in Engineering and Technology*, vol. 2, no. 2, pp. 34–39, 2015.

[279] V. Grabe, H. H. Bülthoff, D. Scaramuzza, and P. R. Giordano, "Nonlinear ego-motion estimation from optical flow for online control of a quadrotor uav," *The International Journal of Robotics Research*, vol. 34, no. 8, pp. 1114–1135, 2015.

Published list

1) P. Righettini, R. Strada, S. Valilou, E. Khademolama, "Output feedback sliding mode controller with H2 performance for robot manipulator" IEEE International Conference on Automation and Computing (ICAC), 2015 21st, pp. 1-6, 11-12 September 2015.

2) P. Righettini, R. Strada, E. Khademolama, S. Valilou, "Symbolic kinematic and dynamic modelling toolbox for Multi-DOF robotic manipulators" IEEE International Conference on Automation and Computing (ICAC), pp. 1-7, 11-12 September 2015.

3) P. Righettini, R. Strada, S. Valilou, E. Khademolama, "Nonlinear modeling and experimental validation of uni-axial servo hydraulic shaking table" International Conference of Bath/ASME on Fluid Power and Motion Control, 2016, 7-9 September.

4) P. Righettini, R. Strada, S. Valilou, E. Khademolama, "Gray-box acceleration modeling of an electro hydraulic servo shaking table with Neural Network" IEEE International Conference on Advanced Intelligent Mechatronics, Munich, 2017, 3-7 July.

Submitted papers

1) P. Righettini, R. Strada, E. Khademolama, S. Valilou, "Online Wavelet Complementary Velocity Estimator" ISA Transactions