DYNAMICS AND TWO NEW PD-TYPE CONTROLLERS FOR ROBOT MANIPULATORS

BY

Jose Alejandro RUEDA MEZA

A Thesis
Submitted to the Faculty of Graduate Studies
in Partial Fulfillment of the Requirements
for the Degree of

MASTER OF SCIENCE

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg, Manitoba

ISBN 0-315-77037-6

DYNAMICS AND TWO NEW PD-TYPE CONTROLLERS

FOR ROBOT MANIPULATORS

BY

JOSE ALEJANDRO RUEDA MEZA

A thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

To my Mother Lucia

and my Grandmother Josefina

for their love and encouragement.

ABSTRACT

The central topic for this thesis is modelling and control of robot manipulators.

A kinematic analysis is presented, based on the Denavit–Hartenberg representation. As a result of this analysis, some important expressions are developed that are later applied in the dynamical modelling of robot manipulators. Important properties of the model , which are exploited in control system analysis are presented.

An original proof is presented for the stability of the so called PD plus computed feedforward compensation controller, that has shown to have excellent performance in experiments.

A new stable PD–type adaptive control law is presented, for independent–joint control of manipulators, with parametric uncertainties in the actuators. We show in simulations that this controller has a very good performance and no case was founded in which the parameters were not identified.

Acknowledgements.

The author would like to thank the Department of Electrical and Computer Engineering of the University of Manitoba, in particular Dr. Witold Pedrycz for his academic support and advices that gave form to this document.

An acknowledgement is extended to Ms. Lucinda Leskie who read the manuscript and made important observations.

The author is deeply grateful to Ms. Shuang Yu for her encouragement and kindness in helping with the preparation of the final version.

An acknowledgment is presented to Dr. Rafael Kelly from Instituto Technologico y de Estudios Superiores de Monterrey, Mexico, who guided the author in the study of control and robotics, and provided him with important concepts, ideas and methods. Without his academic support this work could not have been done.

The author thanks Mr. Efrain Alcorta for his collaboration in the study of feedforward controllers, development and testing of the adaptive controller, and for his friendship.

Finally, the author acknowledges and thanks the Consejo Nacional de Ciencia y Tecnologia (CONACyT), Mexico, for their sponsorship.

J. A. RUEDA M.

CONTENTS

LIST OF FIGURES

0. INTRODUCTION.

INTRODUCTION.

There are several disciplines involved in the study of robotics, namely mechanical engineering, computer science, electronics, mathematics, physics and control engineering[3]. This work has been written from the point of view of control engineering. In order to design a control system that will accomplish a desired task with the use of a robot manipulator, we need a design–specific controller [1,2,3,4,16]. From the theoretical point of view, it is very important to have access to the model of the manipulator we are dealing with. This is the first motivation for the study of robot dynamics. There are several approaches that can be used to develop useful expressions for the dynamics of the robot. From all these techniques for modelling, the Euler–Lagrange approach [4,16] provides us with a model with some very useful properties [4,9,16] that allow us to use powerful tools for the analysis of the closed–loop system, for instance functional analysis [5] and Lyapunov's stability theories [12,17]. This model have been developed in the past years and it have been used to design control algorithms, from PD–type [1,7,11,13] to adaptive [4,6,8,14,15], based primarily on those properties.

THE PROBLEM.

There are several experimental results confirming the theory of robot controllers. Particularly, the PD plus computed feedforward compensation controller has been used in experiments with excelent results, unfortunately no stability proof has been presented. We develop the proof in this document.

The motivation that originated these kinds of controllers is that, although recent technological advances and reducing costs in the field of digital electronics, have permitted the employment of microprocessor–based equipment, with high speed and powerful computations, in control of robot manipulators, it is of great interest the use of control techniques that include a reduced number of operations to be done on–line. Among these control techniques are computed feedforward and PD plus computed feedforward. The main advantage of these controllers is that all the feedforward terms can be calculated off–line.

In [11] a PD controller with cancellation of gravity was proposed, using a precomputed feedforward gravity compensation. It was shown that the closed–loop system is stable. The PD plus computed feedforward controller [1] is an extension of this work, in which a precomputed compensation is introduced in the system for the overall dynamics of the robot.

Feedforward is a technique in control theory [1,17] that allows us to deal with problems of additive perturbations in the input and parametric uncertainties in the model. Feedforward refers to the addition of precomputed terms to the control action at input of the plant. This technique gives some very good results in adaptive control. A good example is presented here, where an adaptive feedforward compensation has been added to a PD–type controller.

Electric motors are one of the most used energy source in low power industrial applications. We present the analysis of a controller for direct current motors (DC motors), to be used as actuators in robot manipulator joints, where we need great precision in the following of trajectory specifications. Brushless motors that emulate via a power electronic interface, the linear transfer function of a DC motor, are widely applied in industrial robots. We are interested in those cases in robotics, in which independent joint control is enough to satisfy control objectives.

There exists a great variety of commercial electronic equipment for DC motors, designated to the applications of velocity regulation, and in less scale for position regulation. This is due to the fact that in the majority of the industrial applications, DC motors are used in tasks that involve motion at constant velocity. However, there are applications in which velocity regulation or position regulation is not enough to satisfy certain specifications of motion. Applications in robotics are a typical examples, where it is required for the manipulator to follow time–varying trajectories of position and velocity. Electronic equipment for these kinds of applications is practically absent in the market, due to the lack of economic feasability.

Although commercial equipment can be successfully used to regulate velocity and sometimes regulate position, they cannot be satisfactorily used in robotics. In this situation, the controller must be specifically designed for the model that characterizes the dynamical behaviour of the system. The situation could be still more complex, when the parameters of the model are difficult to be quantified, of openly unknown. The study of this case was the motive for the development of the present analysis. We use the model of a DC motor presented in [10]. There are three previous solutions to deal with this problem, presented in

a) State Feedback Controller with Adaptive Compensation [6],

b) PD Controller with Adaptive Compensation [14,15], and

c) Adaptive Computed Torque Controller [8].

The simplest of these controllers is the state feedback controller with adaptive compensation. A disadvantage of this controller is that, it is conditioned to an adequate selection of γ. The type of reference functions $q_d(t)$ is also restricted. In the last two controllers this does not happen, but they are much more complex. We propose a simple controller that satisfies the control objective. Although, we do not show convergence of the parametric error to zero, we found in all our simulations, that it does.

This approach is interesting from the application point of view, because we can add the compensation to an existing proportional-derivative controller.

The simulations presented here were made using the simulation package for nonlinear systems SIMNON [18].

ORGANIZATION.

The document is organized as follows.

In order to make the document self-contained, it was decided to include a first chapter with some well-known mathematical results, that are widely used in the analysis or robot control systems. We use this concepts throughout the whole document. In this Chapter 1, we present some important properties from linear algebra, that we will need in the development of the dynamical model of the manipulator in Chapters 2 and 3. We also present some basic concepts and theorems related to Lyapunov stability analysis. At the end of the chapter, we include some definitions about certain important normed spaces, that we use to prove convergence of control algorithms. After a review of important mathematical results, we present a detailed development of the dynamical model of a robot, since this is the starting point for the understanding of its properties and also for the study of problems like robots with flexible links and joints.

Kinematics of manipulators is a starting point for a dynamical analysis. Thus, it was decided to include a basic kinematic analysis in Chapter 2. We start by introducing some general material about homogeneous transformations. We describe the kind of robot manipulator we are considering, and its kinematic configuration, considering positions, velocities, and accelerations. The forward and inverse kinematic problems are also stated. The main result of the chapter is the development of the expressions for the Jacobian matrix. At the end of the chapter, we present some remarks about singularities, and the inverse velocity and acceleration problems.

A good understanding in the dynamics of the manipulators is essential for the analysis of their control systems, that will lead it to perform tasks in the real world. After careful review of the existing literature, the author selected the most important issues related to robot manipulator modelling and elected to put them together in a clear and compact form in Chapter 3. Again, this chapter starts with some general considerations, that have to be well understood, since they are the basis for the development of the dynamical model that is required in the design of robot manipulator control algorithms. We present the dynamical model of a robot manipulator of n degrees of freedom. We introduce the development of the Euler-Lagrange equation of motion, for a system of particles with constraints of motion. We find expressions for the kinetic and potential energy, which we use to define the Lagrangian of the system. Then, we consider the robot manipulator to be a system with constraints of motion of the form described before. This enables us to apply the Euler–Lagrange equation of motion in the development of its dynamical model. With particular expressions for the kinetic and potential energy, we form the Lagrangian of the manipulator, and finally, its equation of motion. At the end of the chapter we mention some important properties of this model.

In Chapter 4, we present the analysis of two motion controllers for robot manipulators: computed feedforward and PD plus computed feedforward. The latter has shown to have an excelent performance in experiments [1]. However, no analysis was presented. We study existence and uniqueness of equilibrium points, stability and achievement of the control objective for the overall control system. The chapter is organized as follows. In Section 4.2, we present the dynamical model of a robot manipulator with n degrees of freedom (dof), rigid links, and ideal actuators. Using some important properties of the model, we rewrite the dynamical equation in terms of the state vector. Then, the problem formulation is stated in Section 4.3. The analysis of the computed feedforward controller is presented in section 4.4. In Section 4.5, we present the analysis of the PD plus computed feedforward controller. We present simulation results in Section 4.6, for the manipulator of one degree of freedom (dof). We give some conclusions in Section 4.7, and the references are listed at the end of the chapter.

In Chapter 5, we present the design of an adaptive controller for direct current motors, with parametric uncertainties, that can be applied to robot manipulators. We show stability and achievement of the control objective for the overall control system. The chapter is organized as follows. In Section 5.2, we present the dynamical model of a DC motor controlled by armature. In Section 5.3, we state the control problem. In Section 5.4, we mention some previous reported solutions. We present the control and adaptation algorithms, and the stability analysis in Section 5.5. In Section 5.6, we present some simulation results. We give some conclusions in Section 5.7, and the references are listed at the end of the document.

The reader interested in kinematics with a good background in linear algebra can directly to Chapter 2. For a more detailed discussion see [3,17]. The reader interested in the dynamics of the robot can go directly to Chapter 3. The reader interested in control only and with a good knowledge of Lyapunov theory will not have problems to understand Chapters 4 and 5, however we recommend to read the review in Chapter 1.

REFERENCES.

[1] An, H., Atkeson, C., Hollerbach, J., *Model-based Control of a Manipulator*, The MIT press, , 1988.

[2] Asada, H., and Slotine, J., *Robot Analysis and Control*, John Wiley and Sons, New York, 1986.

[3] Craig, J., *Introduction to Robotics Mechanics and Control*, Addison-Wesley Publishing Co., Reading, MA, 1986.

[4] Craig, J., *Adaptive Control of Mechanical Manipulators*, Addison-Wesley Publishing Co., Reading, MA, 1986.

[5] Desoer, C., Vidyasagar, M., *Feedback Systems: Input-Output Properties*, Academic Press, New York, 1975.

[6] Kelly, R., "A Linear-State Feedback plus Adaptive Feedforward Control for DC Servo-Motors", IEEE Trans. on Industrial Electronics, vol. IE-34, no. 2, May, 1987.

[7] Kelly, R., *Control de Movimiento de Robots Manipuladores*, DEPFI-UNAM, D-79, Mexico City, 1989.

[8] Kelly, R., "Adaptive Computed Torque plus Compensation Control for Robot Manipulators", Mechanism and Machine Theory, vol. 25, no 2, 1990.

[9] Khosla, P., Kanade, T., "Parameter Identification of Robot Dynamics", Proceedings of the 24th IEEE Conference on Decision and Control, Fort Lauderdale, FL, December, 1985, pp. 1754-1760.

[10] Kuo, B., *Automatic Control Systems*, 4th edition, Prentice Hall, New York, 1982.

[11] Khorrami, F., Özgüner, U., "Decentralized Control for Robot Manipulators via State and Proportional-Integral Feedback", Procc. of the IEEE International Conference on Robotics and Automation, April 24-28, 1988, Philadelphia, PA, vol. 2, pp. 1198-1203.

[12] LaSalle, J. P., *The Stability and Control of Discrete Processes*, Springer Verlag, New York, 1986.

[13] Paden, B., Panja, R., "Globally Asymptotically Stable PD+ Controller for Robot Manipulators", Int. Journal of Control, vol. 47, No. 6, 1988, pp. 1697–1712.

[14] Slotine, J., Li., W., "On the Adaptive Control of Robot Manipulators", The International Journal of Robotics Research, vol. 6, No. 3, Fall, 1987.

[15] Slotine, J., Li., W., "Adaptive Manipulator Control: a Case of Study", IEEE Transactions on Automatic Control, vol. 33, no. 11, November 1988.

[16] Spong, M., and Vidyasagar, M., *Robot Dynamics and Control*, John Wiley & Sons, New York, 1989.

[17] Vidyasagar, M., *Nonlinear Systems Analysis*, Prentice Hall, New Jersey, 1978.

[18] Elmqvist, H., *SIMNON: An Interactive Simulation Program for Nonlinear Systems*, Proceedings Simulation '77, Montreux, 1977.

1. PRELIMINARIES.

In this chapter we present some important properties from linear algebra, that we will need in the development of the dynamical model of the manipulator in Chapters 2 and 3. We also present some basic concepts and theorems related to Lyapunov stability analysis. At the end of the chapter, we include some definitions about certain important normed spaces, that we use to prove convergence of control algorithms.

1.1 LINEAR ALGEBRA.

We present some basic concepts of linear algebra, that we will use in this work. First, we denote the set of real numbers as $\mathbf{R}$, and the set of real vectors as $\mathbf{R}^n$. We are going to work in the vector space $(\mathbf{R}^{nxn},\mathbf{R})$, which denotes the vector space of all real nxn matrices, over the field of the real numbers, with the conventional matrix addition and scalar multiplication. It is well known that the real numbers form a field with the standard addition and multiplication. From the definition of vector space, we have that $(\mathbf{R}^{nxm},\mathbf{R})$ is closed, associative, distributive and commutative under matrix addition and multiplication, and there exist the elements 0 and 1 in $\mathbf{R}$, and $\mathbf{0}$ and $\mathbf{I}$ in $\mathbf{R}^{nxn}$, called the zero and the identity. We also use the subset $(\mathbf{R}^n,\mathbf{R})$ of $(\mathbf{R}^{nxm},\mathbf{R})$, which is closed, and therefore, a subspace under vector addition and scalar multiplication. We use lowercase boldface letters to indicate vectors in $\mathbf{R}^n$, and uppercase letters to denote matrix. All the vectors are assumed to be column vectors, unless otherwise stated. Then, the notation $\mathbf{x} \in \mathbf{R}^n$, means

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix}, \qquad x_i \in \mathbf{R}$$

The *Euclidean norm* of a vector is defined as

$$\|\mathbf{x}\| = \left(x_1^2 + x_2^2 + \cdots + x_n^2\right)^{1/2} = <\mathbf{x},\mathbf{x}>^{1/2}$$

The *inner product* of two vectors in $\mathbf{R}^n$ is defined as

$$< \mathbf{x}, \mathbf{y} > \; = \mathbf{x}^{\mathrm{T}}\mathbf{y} = x_1y_1 + x_2y_2 + \cdots + x_ny_n$$

With these two operations, the subspace ($\mathbf{R}^n$,$\mathbf{R}$) becomes an inner product space. The *outer product* of two vectors in $\mathbf{R}^n$ is defined as

$$\mathbf{x}\mathbf{y}^{\mathrm{T}} = \begin{bmatrix} x_1y_1 & \cdots & x_1y_n \\ x_2y_1 & \cdots & x_2y_n \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ x_ny_1 & \cdots & x_ny_n \end{bmatrix}$$

The inner product and the outer product are related by

$$< \mathbf{x}, \mathbf{y} > \; = \mathbf{x}^{\mathrm{T}}\mathbf{y} = Tr(\mathbf{x}\mathbf{y}^T)$$

Where the function $Tr(.)$ denotes the *trace* of a matrix, and it is defined as the sum of the diagonal elements of the matrix. The following definitions will be quite important in our analysis.

Def. Orthogonal matrix. The square matrix $A=(a_{ij})\in\mathbf{R}^{n\times n}$ is said to be an orthogonal matrix if and only if $A^TA=I$.

Consider the orthogonal matrix $R=[\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{r}_3]^T\in\mathbf{R}^{3\times 3}$, where $\mathbf{r}_i\in\mathbf{R}^{1\times 3}$, for i=1,2,3. Then, the following relationships follow from the definition

$$\|\mathbf{r}_1\| = \|\mathbf{r}_2\| = \|\mathbf{r}_3\| = 1$$

$$< \mathbf{r}_1, \mathbf{r}_2 > \; = \; < \mathbf{r}_1, \mathbf{r}_3 > \; = \; < \mathbf{r}_2, \mathbf{r}_3 > \; = 0$$

Def. Symmetric matrix. A square matrix $A=(a_{ij})\in\mathbf{R}^{n\times n}$ is said to be symmetric if it is equal to its transpose, i.e. $A=A^T$, in other words $a_{ij}=a_{ji}$.

Def. Positive definite matrix. Let $A=(a_{ij})\in\mathbf{R}^{n\times n}$, and A^T=A. Then, A is called *positive definite matrix* if and only if $\mathbf{x}^TA\mathbf{x} > 0$, for every n–dimensional real column vector $[x_1\ x_2\ \ldots\ x_n]^T\neq\mathbf{0}$.

An useful inequality presented in [3] (pp. 161) is $\mathbf{x}^TA\mathbf{x} \geq \lambda_{\min}(A)\mathbf{x}^T\mathbf{x} > 0$, where $\lambda_{\min}(A)$ is the minimum eigenvalue of A.

The following theorem guarantees the existence of the inverse of a positive definite matrix.

Theorem 1.1 ([3] pp. 163) A real symmetric matrix is positive definite if and only if $A=P^TP$, where P is nonsingular.

It follows that since P is nonsingular, $det(P)\neq 0$, then $det(A)=det(P^TP)=det(P^T)det(P)\neq 0$.

Def. Skew symmetric matrix. A square matrix $A=(a_{ij})$ is said to be skew symmetric if it is equal to the additive inverse of its transpose, i.e. $A=-A^T$, in other words $a_{ij}=-a_{ji}$.

Let us consider the set of all 3x3 skew symmetric matrices. Let S be a 3x3 skew symmetric matrix. By definition we have that

$$S^T + S = 0$$

or

$$s_{ji} + s_{ij} = 0, \qquad i,j = 1,2,3$$

Consider $j=i$, then $s_{ii}=0$, for $i=1,2,3$. Thus S has the following form

$$S = \begin{bmatrix} 0 & -s_1 & s_2 \\ s_1 & 0 & -s_3 \\ -s_2 & s_3 & 0 \end{bmatrix} \tag{1.1}$$

Then, for a given vector $\mathbf{a}=[a_1\ a_2\ a_3]^T$, we can define a skew symmetric matrix $S(\mathbf{a})$ as

$$S(\mathbf{a}) = \begin{bmatrix} 0 & -a_1 & a_2 \\ a_1 & 0 & -a_3 \\ -a_2 & a_3 & 0 \end{bmatrix} \tag{1.2}$$

which will be quite important in the kinematic analysis of a manipulator. The operator $S(\mathbf{a})$ is linear, that is

$$S(x\mathbf{a} + y\mathbf{b}) = xS(\mathbf{a}) + yS(\mathbf{b}) \tag{1.3}$$

for scalars x and y. Also for any vector $\mathbf{b}$, we have

$$S(\mathbf{a})\mathbf{b} = \mathbf{a} \times \mathbf{b} = \begin{bmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{bmatrix} \tag{1.4}$$

where **axb** denotes the cross product. This property will be used often.

Consider the vectors $\mathrm{a},\mathrm{b} \in \mathbf{R}^3$, and the orthogonal matrix $R \in \mathbf{R}^{3\mathrm{x}3}$, then

$$R(\mathbf{a} \times \mathbf{b}) = R\mathbf{a} \times R\mathbf{b} \tag{1.5}$$

which only holds for orthogonal matrices.

From the last two equations. We can write

$$RS(\mathbf{a})R^T\mathbf{b} = R\left(\mathbf{a} \times R^T\mathbf{b}\right) = (R\mathbf{a}) \times (RR^T\mathbf{b}) = (R\mathbf{a}) \times \mathbf{b} = S(R\mathbf{a})\mathbf{b} \tag{1.6}$$

Consider now, that the orthogonal matrix R is a function of a single variable x. Since $R(x)$ is orthogonal, we have

$$R(x)R^T(x) = I$$

Taking the derivative, we get

$$\frac{d}{dx}R(x)\ R^T(x) + R(x)\ \frac{d}{dx}R^T = 0$$

We define the matrix S as

$$S = \frac{d}{dx}R(x)\ R^T(x)$$

its transpose is given by

$$S^T = R(x)\ \frac{d}{dx}R^T$$

Now, substituting S and S^T into the above equation, we get

$$S + S^T = 0$$

In other words the matrix S is skew symmetric. From the definition of S, we multiply both sides by $R(x)$ to get

$$\frac{d}{dx}R(x) = SR(x) \tag{1.7}$$

This equation will be very important in later chapters.

A final note in linear algebra is the following relation to obtain the determinant of a partitioned matrix, into square matrices of the same size, that is

$$\det\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \det(A)\det(D - CA^{-1}B) \tag{1.8}$$

In the next section we present some basic concepts and theorems related to Lyapunov stability theory, which are important in the stability analysis of control systems.

1.2 STABILITY IN THE SENSE OF LYAPUNOV.

Lyapunov stability theory is a very important tool for the stability analysis of control systems. This theory deals with the study of the stability properties of the equilibrium of differential equations. In the following we present the basic definitions of stability, and some important theorems, where we refer the reader to [4] for the proofs. We asume that the system is described by the vector differential equation

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t)) \tag{1.9}$$

where $\mathbf{x}(t) \in R^n$ and $\mathbf{f} : R_+ \times R^n \to R^n$. We assume that the function **f** is such that the the differential equation (1.9) has a unique solution.

Equilibrium definitions.

Def. Equilibrium. The vector $\mathbf{x}_o \in R^n$ is said to be an equilibrium point of the system (1.9) at time t_o, if

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}_o) \equiv \mathbf{0}, \qquad \forall t \geq t_o$$

Without loss of generality we assume that the the vector $\mathbf{x}_o=\mathbf{0}$ is the equilibrium point of the system (1.9), since we can always move the origin of the coordinate system of the state space.

Def. Stable equilibrium. The equilibrium point $x_o=\mathbf{0}$ at time t_o of the system (1.9) is said to be stable at time t_o if, for each $\epsilon > 0$, there exists a $\delta > 0$ such that

$$\| \mathbf{x}(t_o) \| < \delta \Rightarrow \| \mathbf{x}(t) \| < \epsilon \qquad \forall t \geq t_o$$

In other words, the equilibrium of the system (1.9) is stable if for each circle of radius ϵ centered in the equilibrium, we can keep the states trajectories inside for $t>0$, whenever the trajectory at time t_o starts inside another circle of radius δ, centered at the equilibrium. If we start the system in the equilibrium,.the state trajectory is a single point, the equilibrium. This is true for every equilibrium. Equivalently, **f=0** implies that the derivative of **x** is zero, which implies that there is no motion of the state.

Lemma ***1.1*** ([4] pp.139) Suppose the equilibrium point **0** at time t_o of the system

(1.9) is stable at some time $t_1 > t_o$. Then **0** is also a stable equilibrium point at all time $\tau \in [t_o, t_1]$.

This lemma says that a stable equilibrium is always a stable equilibrium.

Def. Asymptotically stable equilibrium. The equilibrium point x_o=**0** at time t_o of the system (1.9) is said to be asymptotically stable at time t_o if it is stable, and there exists a number $\delta_1 > 0$ such that

$$\| \mathbf{x}(t_o) \| < \delta_1 \Rightarrow \| \mathbf{x}(t) \| \rightarrow 0 \qquad \text{as } t \rightarrow \infty$$

In other words, the equilibrium of the system (1.9) is asymptotically stable if it is stable, and the state trajectory converges to the equilibrium, whenever we start the system inside a circle of radius δ_1.

Def. Globally asymptotically stable equilibrium. The equilibrium point x_o=**0** at time t_o of the system (9) is said to be globally asymptotically stable at time t_o, if it is stable, and it is attractive for every $\mathbf{x}_o \in R^n$, that is

$$\| \mathbf{x}(t) \| \rightarrow 0 \qquad \text{as } t \rightarrow \infty, \qquad \forall \mathbf{x}_o \in R^n$$

In other words, the equilibrium of the system (1.9) is globally asymptotically stable regardless of the initial condition of the state trajectory, it always converges to the equilibrium. From this definition we can see that an equilibrium has to be unique to be globally asymptotically stable, since by definition, if there were another equilibrium and we start the system in one of them , the system will stay there, hence it will not converge always to the origin.

Functions.

The following definitions about functions are very important in Lyapunov stability theory.

Def. Function of class *K*. A continuous function $w : R \rightarrow R$ is said to belong to class K if

(*i*) w is nondecreasing.
(*ii*) $w(0) = 0$, and
(*iii*) $w(x) > 0, \ \forall x > 0$.

Def. Locally positive definite function. A continuous function $V : R_+ \times R^n \rightarrow R$ is said to be a locally positive definite function, if there exists a function w of class K, such that

$$(i)\ V(t, \mathbf{0}) = 0,\ \forall t \geq 0$$
$$(ii)\ V(t, \mathbf{0}) \geq w(\| \mathbf{x} \|),\ \text{for all } \mathbf{x} \text{ belonging to a ball}$$
$$B_r = \{\mathbf{x} : \| \mathbf{x} \| \leq r\},\ r > 0$$

Lemma 1.2 ([4] pp. 142) A continuous function $W : R^n \rightarrow R$ is said to be locally positive definite function if and only if

$$(i)\ W(\mathbf{0}) = 0$$
$$(ii)\ W(\mathbf{X}) > 0,\ \forall\ \mathbf{x} \neq \mathbf{0}$$
belonging to a ball B_r, $r > 0$.

Def. Positive definite function. A continuous function $V : R_+ \times R^n \rightarrow R$ is said to be a positive definite function, if there exists a function w of class K, such that

$$(i)\ V(t, \mathbf{0}) = 0,\ \forall t \geq 0$$
$$(ii)\ V(t, \mathbf{0}) \geq w(\| \mathbf{x} \|),\ \forall \mathbf{x} \in R^n$$
$$(iii)\ w(p) \rightarrow \infty \text{ as } p \rightarrow \infty$$

Lemma 1.3 ([4] pp. 142) A continuous function $W : R^n \rightarrow R$ is said to be positive definite function if and only if

$$(i)\ W(\mathbf{0}) = 0$$
$$(ii)\ W(\mathbf{X}) > 0,\ \forall\ \mathbf{x} \neq \mathbf{0}$$
$$(iii)\ W(\mathbf{x}) \rightarrow \infty \text{ as } \| \mathbf{x} \| \rightarrow \infty, \text{ uniformly in } \mathbf{x}$$

Lemma 1.4 ([4] pp. 143) A continuous function $V : R_+ \times R^n \rightarrow R$ is a locally positive definite function if and only if there exist a locally positive definite function $W : R^n \rightarrow R$ such that

$$V(t, \mathbf{x}) \geq W(\mathbf{x}),\ \forall t \geq 0,\ \forall \mathbf{x} \in B_r,\ r > 0$$

where B_r is a ball in R^n.

Lemma 1.5 ([4] pp. 143) A continuous function $V : R_+ \times R^n \rightarrow R$ is a positive definite function if and only if there exist a positive definite function $W : R^n \rightarrow R$ such that

$$V(t,\mathbf{x}) \ \geq \ W(\mathbf{x}), \ \forall t \geq 0, \ \forall \mathbf{x} \in R^n$$

Def. Decrescent function. A continuous function $V : R_+ \times R^n \rightarrow R$ is said to be a decrescent function, if there exists a function w of class K, such that

$$V(t,\mathbf{x}) \ \leq \ w(\|\mathbf{x}\|), \ \forall t \geq 0, \ \forall \mathbf{x} \in B_r, \ r > 0$$

where B_r is a ball in R^n.

In other words, the function V is decrescent if for each p in an interval $(0,r)$, we have

$$\sup_{\|\mathbf{x}\| \leq p} \ \sup_{t \geq 0} V(t,\mathbf{x}) \ < \ \infty$$

Lyapunov direct method.

There are two methods in Lyapunov's stability theory, the so called indirect and direct methods, or first and second methods. The first method allows us to draw conclusions about a nonlinear system by studing the behaviour of a linear system, which is the linearization of the nonlinear around the equilibrium point. The second method is based on the use of theorems that provide sufficient conditions for stability of nonlinear system. In this work we are interested in the second method. We shall now present some important theorems.

T**heorem 1.2 **Stability. ([4] pp. 148) The equilibrium point **0** at time t_o of (1.9) is stable if there exists a continuously differentiable locally positive definite function $V(t,\mathbf{x}) : R_+ \times R^n \rightarrow R$ such that

$$\dot{V}(t,\mathbf{x}) \ \leq \ 0, \ \forall t \geq t_0, \ \forall \mathbf{x} \in B_r, \ r > 0$$

where B_r is a ball in R^n.

In other words if we can find a locally positive definite function or a positive definite function of t and $\mathbf{x}$, such that its derivative with respect to time is less or equal than 0, for all $t > t_o$, then the equilibrium of the system (1.9) is stable.

Theorem 1.3 Global asymptotic stability. ([4] pp. 154) The equilibrium point **0** at time t_o of (1.9) is asymptotically stable if there exists a continuously differentiable positive definite function $V(t,\mathbf{x}) : R_+ \times R^n \rightarrow R$ and a function w of class K such that

$$\dot{V}(t,\mathbf{x}) \ \leq \ -w(\|\mathbf{x}\|), \ \forall t \geq t_0, \ \forall \mathbf{x} \in R^n$$

Equivalently, the equilibrium of the system (1.9) is globally asumptotically stable if we can find a positive definite function of t and $\mathbf{x}$ such that the negative of its derivative with respect to time is a positive definite function itself. In other words, if the derivative is 0 only at the equilibrium $\mathbf{x=0}$ and it is negative everywhere else.

Notice that these are only sufficient conditions, therefore if we cannot find a suitable function V, we cannot conclude anything.

Def. Lyapunov candidate function. A Lyapunov candidate function is a positive definite function.

Def. Lyapunov function. A Lyapunov function is a positive definite function and the negative of its derivative with respect to time is also a positive definite function.

Another important theorem that can be applied under certain special conditions is stated after the following definition.

Def. Autonomous system. The system of differential equations (1.9) is said to be autonomous if $\mathbf{f}$ does not depend explicitly on t, that is

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \qquad t \geq 0$$

Theorem 1.4 (La Salle) Autonomous systems. ([5] pp. 9) Suppose that the system (1.9) is autonomous. Suppose that $V(\mathbf{x}) : R^n \rightarrow R$ is a continuous differentiable positive definite function and $\dot{V}(\mathbf{x}) \leq 0, \ \forall \mathbf{x} \in R^n$. Define the set

$$S = \left\{ \mathbf{s} \in R^n : \dot{V}(\mathbf{s}) = 0 \right\}$$

If $\dot{\mathbf{s}} = f(\mathbf{s}), \ s \in S$ has as a unique solution $\mathbf{s=0}$, then the origin is a globally asymptotically stable equilibrium of the system (1.9).

Notice that in this theorem we do not require $-\dot{V}(\mathbf{x})$ to be a positive definite function to guarantee global asymptotic stability of the equilibrium.

1.3 L_p SPACES.

In this section we state the definitions of certain normed spaces, that we will use when proving convergence of the state variables of a closed–loop control system.

Def. Set of measure zero. A set S of real numbers is said to be of measure zero, if S contains either a finite or a countably infinite number of elements.

Def. Mesurable function. A function $f(\cdot) : R \to R$ is said to be measurable if it is continuous everywhere except on a set of measure zero.

Def. Space L_2. The space L_2 is defined as the set of all measurable functions $f(\cdot) : [0, \infty) \to R$, such that

$$\int_0^\infty |f(t)|^2 \, dt < \infty$$

Another important space is the set of all bounded functions, presented next.

Def. Space L_∞. The space L_∞ is defined as the set of all measurable functions $f(\cdot) : [0, \infty) \to R$, such that

$$\underset{t \in [0, \infty)}{ess \ sup} |f(t)| < \infty$$

Where essentially bounded means bounded except on a set of measure zero.

Def. Uniformly continuous function. A function $f(\cdot) : R_+ \to R_n$, is said to be uniformly continuous if $f(\cdot) \in L^n_\infty$, and $\dot{f}(\cdot) \in L^n_\infty$.

In this chapter we have presented some important properties, definitions, and theorems that we will recall in later chapters.

BIBLIOGRAPHY.

[1] Craig, J., *Introduction to Robotics Mechanics and Control*, Addison–Wesley Publishing Co., Reading, MA, 1986.

[2] Spong, M., and Vidyasagar, M., *Robot Dynamics and Control*, John Wiley & Sons, 1989.

[3] Usmani, R., *Applied Linear Algebra*, Marcel Dekker Inc., New York, 1987.

[4] Vidyasagar, M., *Nonlinear Systems Analysis*, Prentice Hall, New Jersey, 1978.

[5] LaSalle, J. P., *The Stability and Control of Discrete Processes*, Springer Verlag, New York, 1986.

2. KINEMATICS OF MANIPULATORS.

In this chapter we present the kinematic analysis of a robot manipulator. We start by introducing some general material about homogeneous transformations. We describe the kind of robot manipulator we are considering, and its kinematic configuration, considering positions, velocities, and accelerations. The forward and inverse kinematic problems are also stated. The main result of the chapter is the development of the expressions for the Jacobian matrix. At the end of the chapter, we present some remarks about singularities, and the inverse velocity and acceleration problems.

2.1 ROTATIONS.

Our first problem of kinematics is how to describe the position of a body in terms of a reference coordinate system.

It is well known that a given vector $\mathbf{p}_1$, that represents the distance from the origin of a coordinate system (x_1,y_1,z_1) to the point $\mathbf{p}$, can be referred to another system (x_o,y_o,z_o) by means of a homogeneous transformation. This is a matrix that multiplies a vector and gives as a result the image of that vector in the new coordinate frame. Let us consider two coordinate systems that coincide in their origin, but one of them is rotated with respect to the other, say that (x_1,y_1,z_1) is rotated with respect to (x_o,y_o,z_o). The rotation matrix that relates the frame 1 with the frame 0 is given in terms of the unit vectors of the two systems $(\mathbf{i}_o,\mathbf{j}_o,\mathbf{k}_o)$ and $(\mathbf{i}_1,\mathbf{j}_1,\mathbf{k}_1)$ as follows

$$ {}^1R_o = \begin{bmatrix} \mathbf{i}_1 \cdot \mathbf{i}_o & \mathbf{j}_1 \cdot \mathbf{i}_o & \mathbf{k}_1 \cdot \mathbf{i}_o \\ \mathbf{i}_1 \cdot \mathbf{j}_o & \mathbf{j}_1 \cdot \mathbf{j}_o & \mathbf{k}_1 \cdot \mathbf{j}_o \\ \mathbf{i}_1 \cdot \mathbf{k}_o & \mathbf{j}_1 \cdot \mathbf{k}_o & \mathbf{k}_1 \cdot \mathbf{k}_o \end{bmatrix} $$

where ${}^1R_o$ indicates the rotation matrix, such that when multiplied by a vector in the frame 1 gives as a result the representation of the vector in the frame 0. This matrix is obtained from the fact that a component of a vector can be represented as follows, since $\mathbf{p}_o$ and $\mathbf{p}_1$ are representations of the same vector $\mathbf{p}$

$$p_{ox} = \mathbf{p}_o \cdot \mathbf{i}_o = \mathbf{p}_1 \cdot \mathbf{i}_o = p_{1x}\, \mathbf{i}_1 \cdot \mathbf{i}_o + p_{1y}\, \mathbf{j}_1 \cdot \mathbf{i}_o + p_{1z}\, \mathbf{k}_1 \cdot \mathbf{i}_o$$
$$p_{oy} = \mathbf{p}_o \cdot \mathbf{j}_o = \mathbf{p}_1 \cdot \mathbf{j}_o = p_{1x}\, \mathbf{i}_1 \cdot \mathbf{j}_o + p_{1y}\, \mathbf{j}_1 \cdot \mathbf{j}_o + p_{1z}\, \mathbf{k}_1 \cdot \mathbf{j}_o$$
$$p_{oz} = \mathbf{p}_o \cdot \mathbf{k}_o = \mathbf{p}_1 \cdot \mathbf{k}_o = p_{1x}\, \mathbf{i}_1 \cdot \mathbf{k}_o + p_{1y}\, \mathbf{j}_1 \cdot \mathbf{k}_o + p_{1z}\, \mathbf{k}_1 \cdot \mathbf{k}_o$$

that is

$$\mathbf{p}_o = {}^1R_o\, \mathbf{p}_1 \tag{2.1}$$

or

$$\begin{bmatrix}\mathbf{i}_o & \mathbf{j}_o & \mathbf{k}_o\end{bmatrix}\begin{bmatrix}p_{ox}\\ p_{oy}\\ p_{oz}\end{bmatrix} = \begin{bmatrix}\mathbf{i}_1 \cdot \mathbf{i}_o & \mathbf{j}_1 \cdot \mathbf{i}_o & \mathbf{k}_1 \cdot \mathbf{i}_o\\ \mathbf{i}_1 \cdot \mathbf{j}_o & \mathbf{j}_1 \cdot \mathbf{j}_o & \mathbf{k}_1 \cdot \mathbf{j}_o\\ \mathbf{i}_1 \cdot \mathbf{k}_o & \mathbf{j}_1 \cdot \mathbf{k}_o & \mathbf{k}_1 \cdot \mathbf{k}_o\end{bmatrix}\begin{bmatrix}\mathbf{i}_1 & \mathbf{j}_1 & \mathbf{k}_1\end{bmatrix}\begin{bmatrix}p_{1x}\\ p_{1y}\\ p_{1z}\end{bmatrix}$$

By the same procedure, we can write the components of $\mathbf{p}_1$ in terms of the components of $\mathbf{p}_o$ and get $\mathbf{p}_1 = {}^oR_1\, \mathbf{p}_o$, where

$${}^oR_1 = \begin{bmatrix}\mathbf{i}_o \cdot \mathbf{i}_1 & \mathbf{j}_o \cdot \mathbf{i}_1 & \mathbf{k}_o \cdot \mathbf{i}_1\\ \mathbf{i}_o \cdot \mathbf{j}_1 & \mathbf{j}_o \cdot \mathbf{j}_1 & \mathbf{k}_o \cdot \mathbf{j}_1\\ \mathbf{i}_o \cdot \mathbf{k}_1 & \mathbf{j}_o \cdot \mathbf{k}_1 & \mathbf{k}_o \cdot \mathbf{k}_1\end{bmatrix}$$

Notice that

$$({}^1R_o)^T = \begin{bmatrix}\mathbf{i}_1 \cdot \mathbf{i}_o & \mathbf{i}_1 \cdot \mathbf{j}_o & \mathbf{i}_1 \cdot \mathbf{k}_o\\ \mathbf{j}_1 \cdot \mathbf{i}_o & \mathbf{j}_1 \cdot \mathbf{j}_o & \mathbf{j}_1 \cdot \mathbf{k}_o\\ \mathbf{k}_1 \cdot \mathbf{i}_o & \mathbf{k}_1 \cdot \mathbf{j}_o & \mathbf{k}_1 \cdot \mathbf{k}_o\end{bmatrix} = {}^oR_1$$

Also $\mathbf{p}_o = {}^1R_o\, \mathbf{p}_1 = {}^1R_o\, ({}^oR_1\mathbf{p}_o) = {}^1R_o\, {}^oR_1\mathbf{p}_o$, hence ${}^1R_o = ({}^oR_1)^{-1}$, and the orthogonality of a rotation matrix follows from

$${}^oR_1 = ({}^1R_o)^T = ({}^1R_o)^{-1}.$$

It is also possible to represent the vector $\mathbf{p}_o$ in the system 1, by using the inverse transformation ${}^0R_1$. which always exists. If the two frames coincide, the rotation matrix is the identity matrix.

If $\mathbf{p}_1$ is the representation of a vector $\mathbf{p}_2$ in the frame 1, where $\mathbf{p}_2$ is in the frame 2, then the representation of $\mathbf{p}_2$ in the frame 0 is a composition of the rotations ${}^1R_o$ and ${}^2R_1$, that is

$$\mathbf{p}_o = {}^1R_o\ {}^2R_1\ \mathbf{p}_2$$

We can extend this result by induction to the case of n rotations by the following composition

$$\mathbf{p}_o = {}^1R_o \; {}^2R_1 \; \cdots \; {}^nR_{n-1} \; \mathbf{p}_n \tag{2.2}$$

It is also possible to rotate a frame around a fixed axis represented in that frame. We are not going to make use of this kind of rotation, because we can always compute the rotation matrix from the unit vectors, once the system has been rotated, and then we can calculate the axis and the angle from this matrix as follows

$$\Theta = \cos^{-1}\left(\frac{Tr(R) - 1}{2}\right) = \cos^{-1}\left(\frac{r_{11} + r_{22} + r_{33} - 1}{2}\right)$$

$$\mathbf{k} = \frac{1}{2\sin\Theta}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}$$

where k is the vector given in terms of the original frame. These formulas can be found in [3] (pp.43).

A rotation matrix can also be represented in terms of the Euler angles (θ,ϕ,ψ). These are not very easy to calculate, but they are used. The procedure of rotating a coordinate system with this method is as follows: first rotate the z axis θ angles, then rotate the y axis ϕ angles, and then the x axis ψ angles. In the last two steps we keep the previous rotations.

Another way of representing a rotation matrix by means of angles is by the use of the so called roll, pitch and yaw angles. The procedure is similar to that described with the Euler angles except that the rotations are performed in reverse order.

2.2 HOMOGENEOUS TRANSFORMATIONS.

Let us present some material about homogeneous transformations. Consider a frame c attached to the center of mass of a body, and an inertial frame i, and the orthogonal rotational matrix ${}^cR_i$, from the frame c to the frame i. This matrix satisfies the following relation

$$\left({}^cR_i\right)^{-1} = \left({}^cR_i\right)^T \tag{2.3}$$

The condition for this equation to hold is that ${}^cR_i$ be nonsingular, i.e. it is possible to make bidirectional transformations

$$ {}^cR_i = \left({}^iR_c\right)^{-1} = \left({}^iR_c\right)^T \tag{2.4}$$

The general homogeneous transformation is of the form

$$ {}^cT_i = \begin{bmatrix} n_x & s_x & a_x & d_x \\ n_y & s_y & a_y & d_y \\ n_z & s_z & a_z & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{n} & \mathbf{s} & \mathbf{a} & {}^c\mathbf{d}_i \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} {}^cR_i & {}^c\mathbf{d}_i \\ \mathbf{0} & 1 \end{bmatrix} \tag{2.5}$$

where **n**, **s**, and **a** represent the direction of the positive x_c, y_c, and z_c axes with respect to the inertial frame i, and ${}^c\mathbf{d}_i$ represents the vector from the origin of the inertial frame, to the origin of the frame in the center of mass. ${}^cR_i$ is the rotation matrix, while ${}^c\mathbf{d}_i$ is the translation vector. Provided that ${}^cR_i$ is orthogonal, it is easy to show that its inverse is given by

$$ {}^iT_c = \left({}^cT_i\right)^{-1} = \begin{bmatrix} {}^cR_i^T & -{}^cR_i^T\,{}^c\mathbf{d}_i \\ \mathbf{0} & 1 \end{bmatrix} \tag{2.6}$$

It is clear that

$$ {}^iT_i = I $$

where I is the 4x4 identity matrix.

In order to perform a homogeneous transformation, we have to add the fourth component to any vector, as follows

$$ \mathbf{P}_i = \begin{bmatrix} \mathbf{p}_i \\ 1 \end{bmatrix} $$

We can call this form the homogeneous representation of a vector. For example, if we want to shift the vector $\mathbf{p}_i$, from the inertial frame to the frame in the center of mass, that is located a distance x_o on the x axis without any rotation, we have

$$ {}^cT_i = \begin{bmatrix} 1 & 0 & 0 & x_o \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

which is the translation matrix from c to i. Its inverse, that gives the translation from i to c, is

$$^{i}T_c = \left({}^{c}T_i \right)^{-1} = \begin{bmatrix} 1 & 0 & 0 & -x_o \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

It is easy to verify that for a given vector $\mathbf{p}_i=[x,y,z]^T$, we have

$$\mathbf{p}_c = {}^{i}T_c \ \mathbf{p}_i = \begin{bmatrix} x + x_o \\ y \\ z \\ 1 \end{bmatrix}$$

and that

$$\mathbf{p}_i = {}^{c}T_i \ \mathbf{p}_c = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Notice that the composition of translation vectors is just the sum of them, referred to the same frame by the rotation, that is

$$^{n}\mathbf{d}_o = {}^{1}d_o + {}^{1}R_o \, {}^{2}d_1 + \cdots + {}^{n-1}R_o \, {}^{n}d_{n-1}$$

2.3 TRANSFORMATION OF LINEAR VELOCITIES.

Let us now talk about transformations of velocity vectors. We know from preliminaries, that an orthogonal matrix has the following derivative with respect to time

$$\dot{R}(t) = S(t) \ R(t) \tag{2.7}$$

where $S(t)$ is a skew symmetric matrix given by

$$S(t) = \dot{R}(t) \ R(t)^T \tag{2.8}$$

This matrix can be represented as $S(\mathbf{w})$ for a unique vector $\mathbf{w}$ [3] (pp. 53) (see preliminaries for details), where $\mathbf{w}$ is the angular velocity of a rotating coordinate system with respect to a fixed one, we have for some vector $\mathbf{p}$ that is

$$S(\mathbf{w})\mathbf{p} = \mathbf{w} \ \mathrm{x} \ \mathbf{p} \tag{2.9}$$

Time–invarying case.

Let $\mathbf{p}_1$ be a constant vector, consider that frame 1 is rotated with respect to the frame 0, and let $\mathbf{p}_o$ to be the image of $\mathbf{p}_1$, in such a way that we obtain $\mathbf{p}_o$ by means of a rotational matrix $R(t)$, that is $\mathbf{p}_o={}^oR_1\mathbf{p}_1$. Let us differentiate $\mathbf{p}_o$ with respect to time, that is

$$\dot{\mathbf{p}}_o = {}^1\dot{R}_o\ \mathbf{p}_1 \qquad (2.10)$$

since $\mathbf{p}_1$ is constant. We know that R is an orthogonal matrix, thus by substituting (2.7) into (2.10) we have

$$\dot{\mathbf{p}}_o = S(\mathbf{w})\ R\ \mathbf{p}_1 = S(\mathbf{w})\ \mathbf{p}_o = \mathbf{w}\ \mathrm{x}\ \mathbf{p}_o \qquad (2.11)$$

where we have assumed that R means ${}^1R_o(t)$. Thus, the linear velocity of the point $\mathbf{p}$ in the frame 0 is given by the cross product of the angular velocity of the frame 1, and the position vector of $\mathbf{p}$ in the frame 0.

Time–varying case.

Consider now that a frame c is in motion with respect to a frame i. In this case the transformation is a function of t, as follows

$${}^cH_i(t) = \begin{bmatrix} {}^cR_i(t) & {}^c\mathbf{d}_i(t) \\ \mathbf{0} & 1 \end{bmatrix} \qquad (2.12)$$

Recall that the homogeneous transformation requires a homogeneous representation of a vector. Let us consider the vector itself in the following manner

$$\mathbf{p}_o = {}^1R_o(t)\ \mathbf{p}_1 + {}^1\mathbf{d}_o(t) \qquad (2.13)$$

The rate of change of the position vector $\mathbf{p}_o$ with respect to time, is given by its velocity, that is

$$\dot{\mathbf{p}}_o = {}^1\dot{R}_o(t)\ \mathbf{p}_1 + {}^1\dot{\mathbf{d}}_o(t) \qquad (2.14)$$

By substituting (2.7) and (2.9) into (2.14), we have

$$\dot{\mathbf{p}}_o = \mathbf{w}\ \mathrm{x}\ {}^1R_o(t)\ \mathbf{p}_1 + {}^o\mathbf{v}_1 \qquad (2.15)$$

where ${}^{1}\mathbf{v}_o = {}^{1}\dot{\mathbf{d}}_o$ is the velocity of the origin of the moving frame with respect to the fixed one. ${}^{o}R_1\mathbf{p}_1$ is the vector from the origin of the moving frame to the extreme of $\mathbf{p}_1$, expressed in the fixed frame. It can be viewed as the radius of the rotation of the point $\mathbf{p}$ (extreme of $\mathbf{p}_1$), with angular velocity $\mathbf{w}$, around the origin of the moving frame at each instant of time. Let us say that $\mathbf{r}={}^{1}R_o\mathbf{p}_1$, and rewrite (2.15) as

$$\dot{\mathbf{p}}_o = {}^{1}\mathbf{v}_o + \mathbf{w} \times \mathbf{r} \tag{2.16}$$

Then, the velocity vector of the point $\mathbf{p}$ with respect to a moving frame 1, and with position vector $\mathbf{r}$, can be expressed as the sum of the linear velocity of the origin of 1 with respect to 0, plus a component due to the relative rotation of 1 with respect to 0.

If $\mathbf{p}_1$ is a time–varying vector, we modify (2.16) as

$$\dot{\mathbf{p}}_o = {}^{1}\mathbf{v}_o + \mathbf{w} \times \mathbf{r} + {}^{1}R_o\, \dot{\mathbf{p}}_1 \tag{2.17}$$

where ${}^{1}R_o\dot{\mathbf{p}}_1$ is the rate of change of $\mathbf{p}_1$ expressed in the fixed frame 0.

Multiple frames.

Let us consider the case in which we have more than two frames, say three. Let us say that $\mathbf{p}_1$ is the representation of $\mathbf{p}_2$ in the frame 1, that is

$$\mathbf{p}_o = {}^{1}R_o \left({}^{2}R_1\, \mathbf{p}_2 + {}^{2}\mathbf{d}_1 \right) + {}^{1}\mathbf{d}_o(t)$$

Now by substituting the derivative of this equation with respect to time into (2.17) we get

$$\dot{\mathbf{p}}_o = {}^{1}\mathbf{v}_o + {}^{1}\mathbf{w}_o \times {}^{1}\mathbf{r}_o + {}^{o}R_1 \left({}^{2}\mathbf{v}_1 1 + {}^{2}\mathbf{w}_1 \times {}^{2}\mathbf{r}_1 + {}^{2}R_1\, \dot{\mathbf{p}}_2 \right)$$

or

$$\dot{\mathbf{p}}_o = {}^{1}\mathbf{v}_o + {}^{1}R_o\,{}^{2}\mathbf{v}_1 + {}^{1}\mathbf{w}_o \times {}^{1}\mathbf{r}_o + {}^{o}R_1\;\; {}^{2}\mathbf{w}_1 \times {}^{2}\mathbf{r}_1 + {}^{2}R_1\, \dot{\mathbf{p}}_2$$

We can extend this result by induction for the case of n frames, as follows

$$\begin{aligned} {}^{n}\dot{\mathbf{p}}_o = {}&{}^{1}\mathbf{v}_o + {}^{1}R_o\,{}^{2}\mathbf{v}_1 + \cdots + {}^{n-1}R_o\;\, {}^{n}\mathbf{v}_{n-1} + \\ &{}^{1}\mathbf{w}_o \times {}^{1}\mathbf{r}_o + {}^{o}R_1\;\, {}^{2}\mathbf{w}_1 \times {}^{2}\mathbf{r}_1 + \cdots + {}^{n-1}R_o\;\, {}^{n}\mathbf{w}_{n-1} \times {}^{n}\mathbf{r}_{n-1} + {}^{n}R_o\,{}^{n}\dot{\mathbf{p}}_{n-1} \end{aligned} \tag{2.18}$$

2.4 *TRANSFORMATION OF ACCELERATIONS.*

Time–invarying case.

It is also possible to express the acceleration of a point $\mathbf{p}$ in a moving coordinate frame, in terms of a fixed frame. Let us obtain this relation by differentiating (2.16)

$$\ddot{\mathbf{p}}_o = {}^1\mathbf{a}_o + \dot{\mathbf{w}} \times \mathbf{r} + \mathbf{w} \times \dot{\mathbf{r}} \tag{2.19}$$

where ${}^1\mathbf{a}_o = {}^1\dot{\mathbf{v}}_o = {}^1\ddot{\mathbf{d}}_o$ is the acceleration of the origin of the moving frame with respect to the inertial frame. Recall that $\mathbf{r}=R\mathbf{p}_1$, then

$$\dot{\mathbf{r}} = \dot{R}\mathbf{p}_1$$

now using (2.7) and (2.9) we have

$$\dot{\mathbf{r}} = S(\mathbf{w})\, R\mathbf{p}_1 = \mathbf{w} \times \mathbf{r}$$

and we can rewrite (2.19) as

$$\ddot{\mathbf{p}}_o = {}^1\mathbf{a}_o + \dot{\mathbf{w}} \times \mathbf{r} + \mathbf{w} \times \left(\mathbf{w} \times \mathbf{r}\right)$$

In other words, the reflected acceleration of a constant vector $\mathbf{p}_1$ has three components. The so called transverse acceleration $\dot{\mathbf{w}} \times \mathbf{r}$, the so called centripetal acceleration wx(wxr), directed towards the axis of rotation, and the linear acceleration of the origin of the moving frame ${}^1\mathbf{a}_o$.

Time–invarying case.

If $\mathbf{p}_1$ is a time–varying vector, we now differentiate (2.17) to get

$$\ddot{\mathbf{p}}_o = {}^1\mathbf{a}_o + \dot{\mathbf{w}} \times \mathbf{r} + \mathbf{w} \times \dot{\mathbf{r}} + R\ddot{\mathbf{p}}_1 + \dot{R}\dot{\mathbf{p}}_1$$

The last term is equal to

$$\dot{R}\dot{\mathbf{p}}_1 = S\, R\dot{\mathbf{p}}_1 = \mathbf{w} \times R\dot{\mathbf{p}}_1$$

also since $\mathbf{r}=R\mathbf{p}_1$

$$\dot{\mathbf{r}} = \dot{R}\mathbf{p}_1 + R\dot{\mathbf{p}}_1 = S\, R\mathbf{p}_1 + R\dot{\mathbf{p}}_1 = \mathbf{w} \times \mathbf{r} + R\dot{\mathbf{p}}_1$$

and

$$\mathbf{w} \times \dot{\mathbf{r}} = \mathbf{w} \times \left(\mathbf{w} \times \mathbf{r}\right) + \mathbf{w} \times R\dot{\mathbf{p}}_1$$

finally

$$\ddot{\mathbf{p}}_o = \mathbf{a} + \dot{\mathbf{w}} \times \mathbf{r} + \mathbf{w} \times \left(\mathbf{w} \times \mathbf{r}\right) + 2\mathbf{w} \times R\dot{\mathbf{p}}_1 \tag{2.20}$$

where $\mathbf{a} = \mathbf{R}\ddot{\mathbf{p}}_1 + {}^1\mathbf{a}_o$. The term $2\mathbf{w} \times R\dot{\mathbf{p}}_1$ is called the Coriolis acceleration.

2.5 TRANSFORMATIN OF ANGULAR VELOCITIES.

Let us consider the composition of two rotation matrices, say ${}^2R_o = {}^2R_1\,{}^1R_o$, and let us differentiate this with respect to time, that is

$$ {}^1\dot{R}_o = {}^1\dot{R}_o\,{}^2R_1 + {}^1R_o\,{}^2\dot{R}_1 \tag{2.21} $$

Let us consider the first term. By using (2.7) and (2.9), it can be modified into

$$ {}^1\dot{R}_o\,{}^2R_1 = S({}^1\mathbf{w}_o)\,{}^1R_o\,{}^2\dot{R}_1 = S({}^1\mathbf{w}_o)\,{}^2R_o \tag{2.22} $$

Now, the second term can be modified by using $RS(\mathbf{a})R^T = S(R\mathbf{a})$ (see preliminaries for details), and the orthogonality of R, that is

$$ \begin{aligned} {}^1R_o\,{}^2\dot{R}_1 &= {}^1R_o\,S({}^2\mathbf{w}_1)\,{}^2R_1 = {}^1R_o\,S({}^2\mathbf{w}_1)\,{}^1R_o + {}^1R_o\,{}^2R_1 \\ &= S({}^1R_o\,{}^2\mathbf{w}_1)\,{}^1R_o\,{}^2R_1 = S({}^1R_o\,{}^2\mathbf{w}_1)\,{}^2R_o \end{aligned} \tag{2.23} $$

Substituting (2.22) and (2.23) into (2.21) yields

$$ {}^2\dot{R}_o = S({}^2\mathbf{w}_o)\,{}^2R_o = \left[S({}^1\mathbf{w}_o) + S({}^1R_o\,{}^2\mathbf{w}_1)\right]\,{}^2R_o $$

Using the linearity of $S(\mathbf{a})$ we have

$$ {}^2\mathbf{w}_o = {}^1\mathbf{w}_o + {}^1R_o\,{}^2\mathbf{w}_1 $$

This equation represents the composition of rotations of angular velocities, when we have two frames. We can extend this result by induction to the case of n frames as follows

$$ {}^n\mathbf{w}_o = {}^1\mathbf{w}_o + {}^1R_o\,{}^2\mathbf{w}_1 + {}^2R_o\,{}^2\mathbf{w}_1 + \cdots + {}^{n-1}R_o\,{}^n\mathbf{w}_{n-1} \tag{2.24} $$

2.6 KINEMATICS OF MANIPULATORS.

General considerations.

All the material described before can be applied to the kinematic analysis of robot manipulators. The kind of robot manipulator that we are considering in this work is an open kinematic chain, composed of rigid links and joints. We represent the motion of the manipulator by joint variables, that are angles in the case of revolute joints, or displacements in the case of sliding joints. We also call the latter prismatic joints.

The revolute joints are found more often in applications. A good reason for this could be that for a straight line workspace, a revolute joint can be the size of the workspace adding the two links together, while the sliding joint has to be at least twice the size of the workspace.

We consider that every joint has only one *degree of freedom*, that means that it only has one joint variable. In the case of joints with two or more degrees of freedom, we can consider them as two or more different joints each with one degree of freedom.

Let us introduce some conventions (see Figure 1 on page 32). Consider a robot with $n+1$ links. We number them from 0 to n, where the link 0 is its base. The links are connected by the joints, we number them from 1 to n. We attach frames to the starting points of the links, such that any motion of the link i will remain as a nonmotion with respect to the same frame i. We denote the joint variables as q_i, where i is the number of the joint. These joint variables are real scalars. We attach the inertial frame to the base, and the other n frames to each link. Hence, the joint variable i is referred to the frame $i-1$. We have a homogeneous transformation matrix between each pair of links. This matrix is a function of the joint variables. This transformation has the following two properties as we said before

$$ {}^{i}T_{i} = I \qquad\qquad {}^{i-1}T_{i} = \left({}^{i}T_{i-1} \right)^{-1} $$

In general, all the equations are valid for an n degrees of freedom robot manipulator. However, it is common to find applications with 6 degrees of freedom, of which 3 are for positioning a hand, and 3 for its orientation. This hand is called the *end-effector*, and from the practical point of view is the most important point of the manipulator. The manipulator performs tasks like grasping, welding, etc. depending on the kind of end-effector that is being used.

The coordinates of the end-effector with respect to the inertial frame, are given by the homogeneous transformation H of the form

$$ {}^{n}H_{o} = \begin{bmatrix} {}^{n}R_{o} & {}^{n}\mathbf{d}_{o} \\ \mathbf{0} & 1 \end{bmatrix} $$

where ${}^{n}R_{o}$ is a composition of rotation matrices, and ${}^{n}\mathbf{d}_{o}$ is a composition of translation vectors. Also every transformation between adjacent links is of the same form, that is

$$ {}^{i}H_{i-1} = \begin{bmatrix} {}^{i}R_{i-1} & {}^{i}\mathbf{d}_{i-1} \\ \mathbf{0} & 1 \end{bmatrix} $$

where i is the number of the joint.

If we need a transformation between frames that are not adjacent, we use composition of rotations and translations.

We already know that the composition of rotations is given by their product, while the composition of translations is given by

$$ {}^{j}\mathbf{d}_{i} = {}^{j-1}\mathbf{d}_{i} + {}^{j-1}R_{i}\,{}^{j}\mathbf{d}_{j-1} \tag{2.25} $$

Depending on the particular configuration of a manipulator, we will have different kinematic representations.

The geometric structure of a manipulator can be represented in different ways. One of the most used is the Denavit–Hartenberg representation, in which we start determining the axis of the joint variable, that we assign to be the z axis, for each joint. Then, we establish the base frame by assigning conveniently x and y to form a right-hand coordinate system. We assign the rest of the coordinate frames in the same way. We then create a table of certain important link parameters that describe its geometry. And finally, we build the n homogeneous transformations. In the case of prismatic joints, we assign the origin of the frame in the beginning of the sliding part. Thus the prismatic link i will have a frame with parallel axes to the frame $i-1$. That is all we are going to say about the Denavit–Hartenberg representation, since we are interested in general descriptions of manipulators.

2.7 POSITION.

During the motion planning process, we have two major objectives in performing a kinematic analysis of a manipulator. First, we want to be able to *find* the end–effector, given some joint variable values, and second, we want to find the joint variables, that will put the end–effector in a desired position. The first of these problems is called the forward kinematic problem, while the second is called the inverse kinematic problem.

Forward kinematics.

The first problem can be solved once we have the homogeneous transformation from the end–effector to the base. The position and orientation of the end–effector is given by the composition of homogeneous transformations of each pair of links. Then, the problem is to find the transformations of each pair of links, multiply them together, and substitute the given values of the joint variables. Recall that each homogeneous transformation between a pair of links, depends only on one joint variable. The problem can be simplified, by choosing an adequate representation, like the Denavit–Hartenberg, In this problem we always have a solution, that is determined simply by straight forward evaluation of the equations.

Inverse kinematics.

The problem is to find a solution to the following equation

$$^{n}T_{o} = H$$

where $^{n}T_{o}$ is the composition of homogeneous transformations of each pair of links, from the link n *to the base,* in terms of the joint variables, for a manipulator of n degrees of freedom. H is the transformation that represents the position and orientation of the end–effector, with respect to the inertial frame, where its position is given by $^{n}\mathbf{d}_{o}$, and its orientation by $^{n}R_{o}$. This results in 16 nonlinear equations, due to the fact that $^{n}T_{o}$ and H are 4x4 matrices. Notice that the bottom row of them is the vector [0,0,0,1] (see (2.7)), this simplifies the problem. Actually, we have to solve 12 algebraic equations that are nonlinear in general, since dependence on the joint variables appears in terms of sines and cosines.

In the inverse kinematic problem, we could have more than one solution or none at all. We are interested in finding a closed–form solution of the equations, instead of a numerical one, so that the trajectory generation system can quickly provide the trajectory, for the controller to follow.

The inverse kinematic problem is, in general, a difficult one, where we have to consider workspace constraints (nonholonomic, see chapter 3 for definition), as well as actuator or link constraints. However, in the case of manipulators with 6 degrees of freedom, from which 3 are for position and 3 for orientation, the problem can be reduced to the solution of the so called inverse position kinematic problem, and the inverse orientation kinematic problem. There exists some algorithms to solve the inverse kinematic problem for this kind of robot. A general solution has not been reported, however, there is a huge amount of research in the area of optimal trajectory generation, that is basically an optimal solution for the inverse kinematic problem.

2.8 VELOCITY.

We now consider velocities. As we described before, we can have transformations of velocities between frames. In the case of robot manipulators there is a special way to represent transformation of velocities. We use a 6xn matrix called the *Jacobian* of the manipulator, in which we put together the transformations for linear and angular velocities. We want to represent these velocities in terms of joint velocities. Suppose that we want the linear and angular velocities of the end–effector in terms of $\dot{\mathbf{q}}$, the derivative of the joint variables. We need expressions of the form

$$
\begin{aligned}
{}^{n}\mathbf{v}_o &= J_v\, \dot{\mathbf{q}} \\
{}^{n}\mathbf{w}_o &= J_w\, \dot{\mathbf{q}}
\end{aligned}
$$

where J_v and J_w are 3xn matrices. Notice that

$$
\begin{aligned}
{}^{n}\mathbf{v}_o &= J_{v1}\dot{q}_1 + J_{v2}\dot{q}_2 + \cdots + J_{vn}\dot{q}_n \\
{}^{n}\mathbf{w}_o &= J_{w1}\dot{q}_1 + J_{w2}\dot{q}_2 + \cdots + J_{wn}\dot{q}_n
\end{aligned}
$$

where J_{vi} and J_{wi} are the column vectors of the matrices J_v and J_w, so that the Jacobian matrices for link i are given by

$$J_v^{(i)} = \left[J_{v1}^{(i)}\, J_{v2}^{(i)} \cdots J_{vi}^{(i)}\ o\ \cdots o\right]$$
$$J_w^{(i)} = \left[J_{w1}^{(i)}\, J_{w2}^{(i)} \cdots J_{wi}^{(i)}\ o\ \cdots o\right]$$

where the column vectors are set to be 0 for $j>i$, because the motion of link i, only depends on joints 1 to i. We write the equations together as

$$\begin{bmatrix} {}^n\mathbf{v}_o \\ {}^n\mathbf{w}_o \end{bmatrix} = {}^nJ_o\,\dot{\mathbf{q}}$$

where

$${}^nJ_o = \begin{bmatrix} J_v \\ J_w \end{bmatrix}$$

is called the Jacobian. In other words, the Jacobian relates infinitesimal joint displacements $d\mathbf{q}$ to infinitesimal end-effector displacements $d\mathbf{x}$, where $\mathbf{x} = [{}^n\mathbf{v}_o\ \ {}^n\mathbf{w}_o]^T$, that is

$$d\mathbf{x} = J\ d\mathbf{q}$$

Recall that the linear velocity of the end-effector is given by

$${}^n\mathbf{v}_o = {}^n\dot{\mathbf{d}}_o$$

where ${}^n\mathbf{d}_o$ is the transformation vector from the joint n to the inertial frame.

The angular velocity of the end-effector is represented by

$$S({}^n\mathbf{w}_o) = {}^n\dot{R}_o\left({}^nR_o\right)^T$$

where ${}^nR_o$ is the rotation matrix from the joint n to the inertial frame.

Angular velocity.

Let us consider the angular velocity of the link i with respect to the link i–1, we can express it as

$${}^i\mathbf{w}_{i-1} = \dot{q}_i\ \mathbf{k}$$

where $\mathbf{k}$ is a unit vector in the z axis of the frame i–1, and q_i is the ith revolute joint variable. Recall that the joint variables are scalar quantities, and ${}^{i}\mathbf{w}_{i-1}$ is a vector with the same direction as the z axis. We mentioned before that there are some conventions in the selection of the coordinate frames at each joint. In this case we say that the z axis is the rotation axis of the link $i-1$. Notice that in order to express the position q_i of joint i with respect to the frame $i-1$, we use a homogeneous transformation that includes rotation and translation, and hence when we talk about angular velocity the constant terms added to the positions are eliminated (see (2.24)), that is, only the rotations affect the representations of an angular velocity with respect to another frame.

Notice that in the last equation we are building a vector in the z_{i-1} axis with magnitude equal to $\dot{q}_i$. Thus, different angular velocities of the link i with respect to the link $i-1$, will be vectors having the same direction but different magnitude.

In the case of sliding joints, the relative angular velocity is zero, because the motion does not depend on an angle.

The angular velocity of the end–effector with respect to the base is a composition of relative angular velocities of each pair of links. We have to consider the case of sliding joints with $\mathbf{w}_i$=0 separately, because our last equation could give an erroneous result using the value of $\dot{q}$ corresponding to the linear displacement. Then, we introduce a constant α_i, that takes the value 0 when the joint i is a sliding joint, and 1 when it is a revolute one. From (2.24) we now have the equation

$$ {}^{n}\mathbf{w}_o = \alpha_1\dot{q}_1\mathbf{k} + \alpha_2\dot{q}_2\,{}^{1}R_o\mathbf{k} + \cdots + \alpha_n\dot{q}_n\,{}^{n-1}R_o\mathbf{k} $$

since the q_i's are scallars. Equivalently

$$ {}^{n}\mathbf{w}_o = \sum_{i=1}^{n}\alpha_i\dot{q}_i z_{i-1} = \left[\alpha_1 z_o \cdots \alpha_n z_{n-1}\right]\dot{\mathbf{q}} $$

where

$$ \alpha_i = \left\{ \begin{array}{c|c} 0 & \text{prismatic joints} \\ 1 & \text{revolute joints} \end{array} \right\} $$

and

$$ z_{i-1} = {}^{i-1}R_o\mathbf{k} \qquad (2.26) $$

It is clear that z_o=$\mathbf{k}$=$[0,0,1]^T$. We now have the value of J_w of the Jacobian as

$$ J_w = \left[\alpha_1 z_o \cdots \alpha_n z_{n-1}\right] \qquad (2.27) $$

Linear velocity.

Now, let us consider the linear velocity of the end-effector. Its value is equal to the change with respect to time of the translation vector from the end-effector to the base, that is

$$ {}^{n}\mathbf{v}_{o} = {}^{n}\dot{\mathbf{d}}_{o} = \sum_{i=1}^{n} \frac{\delta\ {}^{n}\mathbf{d}_{o}}{\delta\ q_{i}}\dot{q}_{i} = \left[\frac{\delta\ {}^{1}\mathbf{d}_{o}}{\delta\ q_{i}} \cdots \frac{\delta\ {}^{n}\mathbf{d}_{n-1}}{\delta\ q_{n}} \right] \dot{\mathbf{q}} $$

Then J_v is

$$ J_{v} = \left[\frac{\delta\ {}^{1}\mathbf{d}_{o}}{\delta\ q_{i}} \cdots \frac{\delta\ {}^{n}\mathbf{d}_{n-1}}{\delta\ q_{n}} \right] \tag{2.28} $$

We can generate the ith column of the Jacobian by keeping all the joints fixed except the ith , which we move at unit velocity q_i=1.

Particular case.

Let us try to find a more simple and systematic expression for J_v, by restricting the kind of manipulator we are dealing with. Consider that our manipulator has its axis x_i intersecting its axis z_{i-1} (see figure 2). In this case, the vector from the origin of the frame $i-1$ to the origin of the frame i, i.e. ${}^{i}\mathbf{d}_{i-1}$, is given by

$$ {}^{i}\mathbf{d}_{i-1} = d_i\mathbf{k} + a_i\,{}^{i}R_{i-1}\mathbf{i} $$

where d_i represents the distance along the z_{i-1} axis, from the origin of $i-1$ to the intersection with x_i, a_i is the distance along the x_i axis from its origin to the intersection with z_{i-1}, and **k** and **i** are unit vectors. The parameters d_i and a_i come from the Denavit–Hartenberg representation, and d_i is the joint variable when the joint $i-1$ is prismatic. We refer the vector $a_i\mathbf{i}$ to the frame $i-1$ by multiplying it by ${}^{i}R_{i-1}$. Notice that whenever we talk about unit vectors, we do not need to specify the frame, because they are all the same magnitude directed towards the axis in any frame. However, we must be careful when we rotate or translate them, we have to multiply them by the corresponding transformation matrix.

Let us keep all the joints fixed except the ith. Differentiating the composition of translations, we have

$$ {}^{n}\dot{\mathbf{d}}_{o} = {}^{i-1}R_{o}\,{}^{i}\dot{\mathbf{d}}_{i-1} $$

The only component in the composition is ${}^{i-1}R_o\,{}^{i}\dot{\mathbf{d}}_{i-1}$, because all the others remain constant.

Prismatic joints.

In a prismatic joint i, ${}^{i}R_{i-1}$ is independent of q_i=d_i, and a_i is a constant. Thus, we have by using above equations and (2.26)

$$ {}^{n}\dot{\mathbf{d}}_o = {}^{i-1}R_o\,{}^{i}\dot{\mathbf{d}}_{i-1} = {}^{i-1}R_o(\dot{d}_i\ \mathbf{k}) = \dot{d}_i\,{}^{i-1}R_o\mathbf{k} = \dot{d}_i z_{i-1} = z_{i-1}\dot{d}_i $$

Also by using the chain rule

$$ {}^{n}\dot{\mathbf{d}}_o = \frac{\delta\ {}^{n}\mathbf{d}_o}{\delta\ q_i}\ \dot{q}_i $$

Hence, since $\dot{q}_i = \dot{d}_i$ is a scalar, we have

$$ \frac{\delta\ {}^{n}\mathbf{d}_o}{\delta\ q_i} = z_{i-1} \tag{2.29} $$

This formula is used to calculate the coefficients of the Jacobian, that corresponds to linear velocities in prismatic joints.

Revolute joints.

Let us consider now revolute joints. The vector from the origin of the inertial frame to the origin of the end–effector's frame is given by

$$ {}^{n}\mathbf{d}_o = {}^{i-1}\mathbf{d}_o + {}^{i-1}R_o\,{}^{n}\mathbf{d}_{i-1} \tag{2.30} $$

from which

$$ {}^{n}\mathbf{d}_o - {}^{i-1}\mathbf{d}_o = {}^{i-1}R_o\,{}^{n}\mathbf{d}_{i-1} \tag{2.31} $$

In the case that only the ith joint is actuated, the vector ${}^{i-1}\mathbf{d}_o$ and the matrix ${}^{i}R_o$ remain constant. Now, differentiating (2.30) with respect to time yields

$$ {}^{n}\dot{\mathbf{d}}_o = {}^{i-1}R_o\,{}^{n}\dot{\mathbf{d}}_{i-1} \tag{2.32} $$

The vector ${}^{n}\dot{\mathbf{d}}_{i-1}$ is generated when the ith joint rotates around the axis z_{i-1}, as we said before. Consider ${}^{n}\mathbf{d}_{i-1} = {}^{n}R_{i-1}\mathbf{p}$, where $\mathbf{p}$ is the representation of ${}^{n}\dot{\mathbf{d}}_{i-1}$ in the frame n. Then by using (2.11)

$$^n\dot{\mathbf{d}}_{i-1} = \mathbf{w}_i \text{ x } {}^n\mathbf{d}_{i-1}.$$

Since the angular velocity of the link i is given by $\mathbf{w}_i = \dot{q}_i\mathrm{K}$, and that the joint variables are scalars, we get

$$^n\dot{\mathbf{d}}_{i-1} = \dot{q}_i\mathbf{k} \text{ x } {}^n\mathbf{d}_{i-1} \tag{2.33}$$

Substituting (2.33) into (2.32) yields

$$^n\dot{\mathbf{d}}_o = {}^{i-1}R_o\left(\dot{q}_i\mathbf{k} \text{ x } {}^n\mathbf{d}_{i-1}\right)$$

which is equal to

$$^n\dot{\mathbf{d}}_o = \dot{q}_i\,{}^{i-1}R_o\,\mathbf{k} \text{ x } {}^{i-1}R_o\,{}^n\mathbf{d}_{i-1}$$

Since R is orthogonal. Recalling (2.26) and using (2.31), we can rewrite this as

$$^n\dot{\mathbf{d}}_o = \dot{q}_i z_{i-1} \text{ x } \left({}^{i-1}\mathbf{d}_o - {}^{i-1}\mathbf{d}_o\right) = \frac{\delta}{\delta q_i}\,{}^n\mathbf{d}_o\dot{q}_i$$

by the chain rule. Therefore,

$$\frac{\delta\,{}^n\dot{\mathbf{d}}_o}{\delta q_i} = z_{i-1} \text{ x } \left({}^{i-1}\mathbf{d}_o - {}^{i-1}\mathbf{d}_o\right) \tag{2.34}$$

This equation represents the ith column of J_v for revolute joints.

Jacobian.

Now, by substituting (2.29) or (2.34), for prismatic joints or revolute joints respectively, into (2.28) we can calculate the ith columns of J_v, while (2.27) gives the value of J_w. This is valid only for the specific configuration that we are considering. Then we have for prismatic joints

$$J_i = \begin{bmatrix} J_{vi} \\ J_{wi} \end{bmatrix} = \begin{bmatrix} z_{i-1} \\ 0 \end{bmatrix} \tag{2.35}$$

and for revolute joints

$$J_i = \begin{bmatrix} J_{vi} \\ J_{wi} \end{bmatrix} = \begin{bmatrix} z_{i-1} \text{ x } \left({}^n\mathbf{d}_o - {}^{i-1}\mathbf{d}_o\right) \\ z_{i-1} \end{bmatrix} \tag{2.36}$$

where J_i is the ith column of the Jacobian. Dependence on $\mathbf{q}$ is easily seen from these equations, because as defined in (2.26), z depends on R, that depends on $\mathbf{q}$, and $\mathbf{d}$ also depends on $\mathbf{q}$, Hence

J is a function of $\mathbf{q}$. Let us write $J(\mathbf{q})$.

We can now calculate the Jacobian of any manipulator that fills our requirements. Notice that z_i is a rotated unit vector, that we can take from the first three rows of the third column of the homogeneous transformation ${}^{i}T_o$ (see (2.26)). The elements ${}^{n}\mathbf{d}_o$ and ${}^{i-1}\mathbf{d}_o$ are the first three elements of the fourth column of the transformations ${}^{i}T_o$ and ${}^{i-1}T_o$ respectively.

The procedure that has been just described is used to compute the velocity of any point of the manipulator in motion. Let us remember, however, that the homogeneous transformation, and therefore the Jacobian, depend on the specific manipulator's configuration.

2.9 ACCELERATION.

We can derive a relation between the joint velocities and accelerations by differentiating the Jacobian. Let us call $\dot{\mathbf{x}}$ the vector of linear and angular velocities of the left–hand side of equation of the Jacobian, that is

$$\dot{\mathbf{x}} = \begin{bmatrix} {}^{n}\mathbf{v}_o \\ {}^{n}\mathbf{w}_o \end{bmatrix} = {}^{n}J_o(\mathbf{q})\ \dot{\mathbf{q}} \tag{2.37}$$

where $\mathbf{x}$ is a vector with 6 elements representing the 3 components of the linear velocity and the 3 components of the angular velocity. By differentiating (2.37), we can obtain a relation between joint accelerations, and linear and angular accelerations of the end–effector. That is

$$\ddot{\mathbf{x}} = J(\mathbf{q})\ \ddot{\mathbf{q}} + \frac{d}{dt}J(\mathbf{q})\ \dot{\mathbf{q}}$$

Singularities.

Suppose that we want to calculate the joint velocities given $\dot{\mathbf{x}}$. This inverse velocity problem can then be solved for the six degrees of freedom manipulator, by solveing the following equation for $\dot{\mathbf{q}}$

$$ {}^{n}J_o(\mathbf{q})\dot{\mathbf{q}} = \dot{\mathbf{x}} \tag{2.38}$$

which has a unique solution if and only if the Jacobian is a nonsingular matrix.

For manipulators with fewer than six links or for redundant manipulators, the does not have inverse, since it is not a square matrix. We can make a full-rank ractorization of the Jacobian of the form $J=EF$, where E is the matrix of the p linearly independen colums of J, and F is the matrix of the q linearly independent rows of the reduced echelon form of J [4] (pp. 96). This factorization is not unique. The Moore-Penrose inverse of J is given by $J^+ = F^+E^+$, where $F^+ = F^T(FF^T)^{-1}$, a right inverse of F, and $E^+ = (E^TE)^{-1}E^T$, a left inverse of E. Thus,

$$\mathbf{q} = {}^nJ_o(\mathbf{q})^+\mathbf{x}$$

However, the joint variables calculated by this equation may not be attainable.

When for some point q the Jacobian is a singular matrix, i.e. $det(J(q))=0$, we say that q is a singular point of the manipulator. Notice that J can only be singular for manipulators with 6 dof (degrees of freedom). These singular points correspond to configurations from which certain directions of motion may be unattainable. Sometimes they represent limits of the manipulator work-space, that is, the points of maximum reach, or points that are unreachable. At singularities, the manipulator would not be able to apply any force over an object with the end-effector. Also there may exist different solutions for the inverse problem at singularities.

If $det(J(q))\neq 0$ then the joint accelerations are given by

$$\ddot{\mathbf{q}} = J(q)^{-1}\left(\ddot{\mathbf{x}} - \frac{d}{dt}J(q)\ \dot{\mathbf{q}}\right)$$

It is important to determine the singularities of a manipulator in order to determine whether or not a solution exists for a certain given task that we want the manipulator to perform.

When controlling a manipulator, the planning system will provide the controller with references to follow, once the inverse problems have been solved.

We will use the Jacobian in the development of the dynamical model of the manipulator in the next chapter.

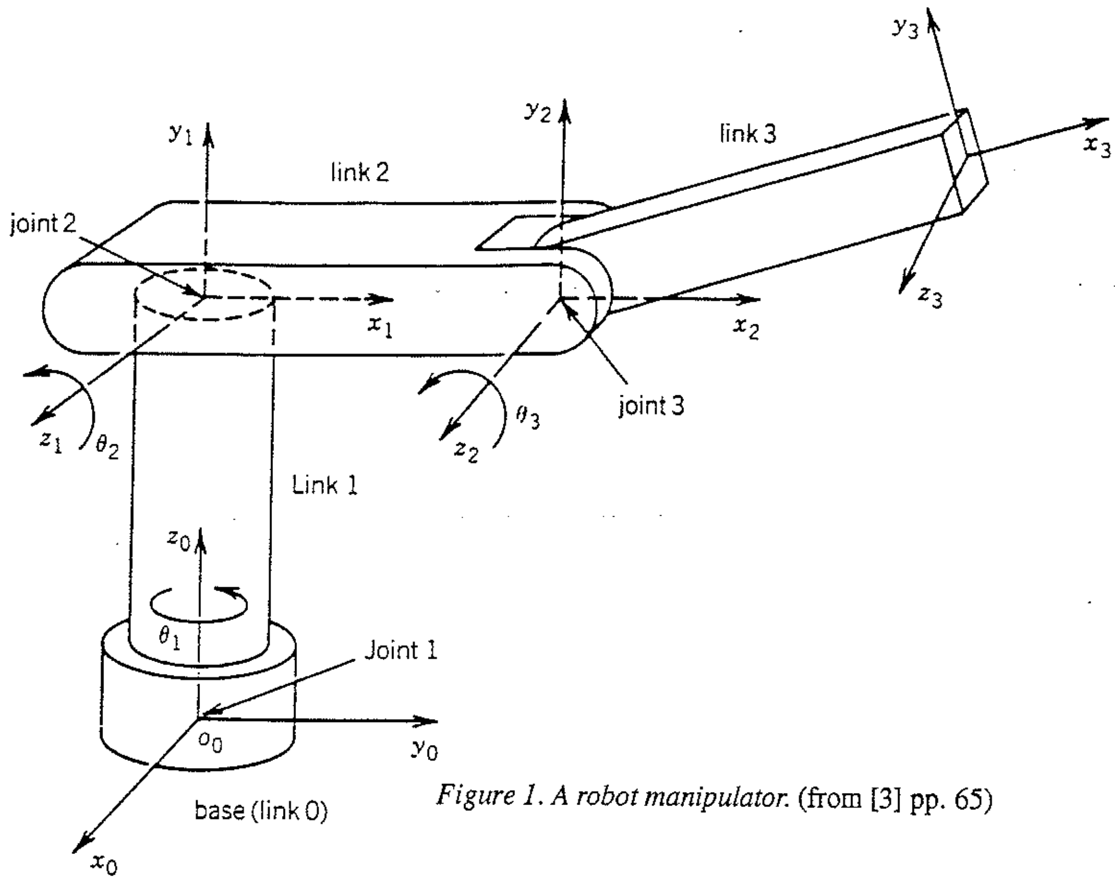


Figure 1. A robot manipulator. (from [3] pp. 65)

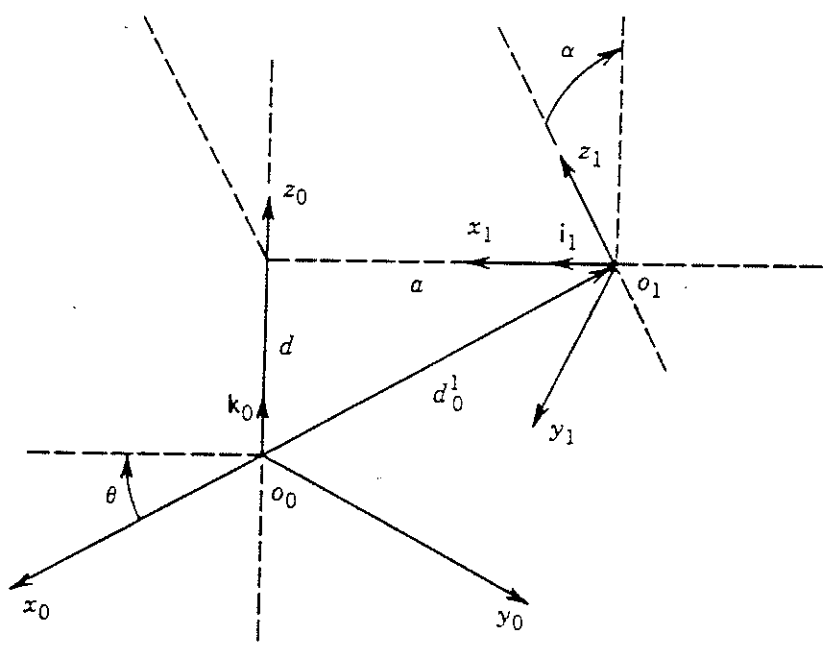


Figure 2. A special case. (from [3] pp. 67)

BIBLIOGRAPHY.

[1] Asada, H., and Slotine, J., *Robot Analysis and Control*, John Wiley and Sons, New York, 1986,

[2] Craig, J., *Introduction to Robotics Mechanics and Control*, Addison–Wesley Publishing Co., Reading, MA, 1986.

[3] Spong, M. and Vidyasagar, M., *Robot Dynamics and Control*, John Wiley and Sons, New York, 1989.

[4] Usmani, R., *Applied Linear Algebra*, Marcel Dekker Inc., New York, 1987.

3. DYNAMICS OF MANIPULATORS.

In this chapter we develop the dynamical model of a robot manipulator of n degrees of freedom. We start with the development of the Euler–Lagrange equation of motion, for a system of particles with constraints of motion. We find expressions for the kinetic and potential energy, which we use to define the Lagrangian of the system. Then, we consider the robot manipulator to be a system with constraints of motion of the form described before. This enables us to apply the Euler–Lagrange equation of motion in the development of its dynamical model. With particular expressions for the kinetic and potential energy, we form the Lagrangian of the manipulator, and finally, its equation of motion. At the end of the chapter we mention some important properties of this model.

3.1 THE EULER–LAGRANGE EQUATION OF MOTION.

Systems of particles.

The Euler–Lagrange equation describes the evolution of a mechanical system subject to holonomic constraints, when these constraint forces satisfy the principle of virtual work. The method presented here is based on the principle of virtual displacements.

Consider a system of k particles, with coordinates $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_k$. If these particles are free to move without any restriction, we can easily describe their motion by Newton's second law. If their motion is now constrained in some fashion, we must consider the constraint forces as well as the externally applied forces. The constraint forces are the forces present in the system, that make the constraint motion hold.

Let us consider a system of 2 particles, in order to illustrate the role of the constraint forces. Suppose that the 2 particles are joined by a massless rigid wire of length l. This wire causes that the following constraint between the two coordinates is satisfied

$$\|\mathbf{r}_1 - \mathbf{r}_2\| = l = \left\langle \mathbf{r}_1 - \mathbf{r}_2, \mathbf{r}_1 - \mathbf{r}_2 \right\rangle^{1/2} = \left[(\mathbf{r}_1 - \mathbf{r}_2)^T\ (\mathbf{r}_1 - \mathbf{r}_2)\right]^{1/2}$$

or

$$(\mathbf{r}_1 - \mathbf{r}_2)^T (\mathbf{r}_1 - \mathbf{r}_2) = l^2 \tag{3.1}$$

If we apply external forces to the particles, they will experiment not only these forces but also the force exerted by the wire. In order to analyze the motion of the two particles, we have two options: a) we could calculate the constraint force due to each set of external forces, b) we could try to determine the motion of the system without any knowledge of the explicit value of the constraint forces at all. These are two approaches in deriving the dynamical equation of motion of a robot manipulator. The first approach is the Newton–Euler formulation and the second is the Lagrangian formulation. The Newton–Euler formulation is derived by the use of Newton's second law of motion, which incorporates forces and moments acting on the particles, including constraint forces due to the coupling. Thus, additional arithmetic operations are required, in order to obtain explicit relations between external forces and displacements. In the Lagrangian formulation, we use relations in terms of work and energy in a generalized coordinate frame. Then, all the constraint forces are eliminated. The resulting equations express in a closed-form, the relations between the external forces and the displacements. The derivation of these equations is simpler than in the Newton–Euler formulation, and it is systematic. Those are the reasons that motivated the use of the Lagrangian formulation in this analysis of systems of particles, with constraints of motion. This technique will be applied later in the dynamical analysis of a robot manipulator, in which we want to express the joint displacements in terms of the torques applied. In the following we will derive the Lagrange's equation of motion. First, we introduce some terminology.

Def. Holonomic constraint. A constraint on the k coordinates $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_k$ is called holonomic if it is an equality constraint of the form

$$g_i(\mathbf{r}_1,\ \mathbf{r}_2,\ \cdots, \mathbf{r}_k) = 0, \qquad i = 1, 2, \cdots, l$$

where the g_i's are the l equations, representing l constraints in the coordinates. In the example of the two particles the function g_i is

$$g_1 = \| \mathbf{r}_1 - \mathbf{r}_2 \| - l = 0$$

In order to illustrate a nonholonomic constraint, consider a particle inside a sphere of radius x, centered at the origin of the coordinate system. The motion of the particle is unconstrained, as long as the particle remains away from the wall of the sphere. It will experiment the constraint force, when it comes into contact with the wall. The constraint in the coordinate of this particle is

$$\|\mathbf{r}\| < x$$

or

$$g_1 = \|\mathbf{r}\| - x < 0$$

Thus, the constraint is nonholonomic, because g_1 is not extrictly equal to zero.

In the case of robot manipulators, the holonomic constraints in the joints are due to the links, while the nonholonomic ones are due to actuator saturation limits, and the workspace, for example. Workspace constraints usually are not considered in the dynamical analysis, however, they are used in generating the trajectory of motion of the manipulator. This is done by a higher stage in the hierarchy of the controller, in the trajectory generation algorithm, that solves the inverse kinematic problem.

A system with l holonomic constraints, may be viewed as having l fewer degrees of freedom, than the unconstrained system. In this case we can express the coordinates of the k particles in terms of *n generalized coordinates* $q_1, q_2, \ldots, q_n$, where $n=k-l$. The coordinates of the particles are then expressed as

$$\mathbf{r}_i = \mathbf{r}_i\,(q_1, q_2, \cdots, q_n), \qquad i = 1, 2, \cdots, k \tag{3.2}$$

where the vectors q_i are linearly independent.

In the case of an infinite number of particles with constraints, we can use a generalized coordinate system. Consider for example, a rigid body that consists of many particles. When the body is in motion, the distances between the particles remain the same. In this case only six coordinates are sufficient, to completely specify the coordinates of any particle in the body. We need three to specify position, and three to specify orientation. The same applies to robot manipulators. A manipulator with more than six degrees of freedom is called redundant. This kind of manipulator is used when dealing with workspace constraints, such as walls. In the rest of the analysis we assume that the number of particles is finite.

Def. Virtual displacement. A virtual displacement is an infinitesimal displacement consistent with the holonomic constraints.

Consider again the example of the two particles constrained by (3.1). Suppose that $\Delta\mathbf{r}_1$ and $\Delta\mathbf{r}_2$ are infinitesimal displacements. The modified coordinates must satisfy this constraint. Then, we have

$$(\mathbf{r}_1 + \Delta\mathbf{r}_1 - \mathbf{r}_2 - \Delta\mathbf{r}_2)^T\ (\mathbf{r}_1 + \Delta\mathbf{r}_1 - \mathbf{r}_2 - \Delta\mathbf{r}_2) = l^2$$

Expanding the product we have

$$(\mathbf{r}_1 - \mathbf{r}_2)^T\ (\Delta\mathbf{r}_1 - \Delta\mathbf{r}_2) + (\Delta\mathbf{r}_1 - \Delta\mathbf{r}_2)^T\ (\mathbf{r}_1 - \mathbf{r}_2) + (\Delta\mathbf{r}_1 - \Delta\mathbf{r}_2)^T\ (\Delta\mathbf{r}_1 - \Delta\mathbf{r}_2) = 0$$

where we have used the fact that (3.1) still holds. Let us neglect the quadratic terms in $\Delta\mathbf{r}_1$ and $\Delta\mathbf{r}_2$, and get

$$(\mathbf{r}_1 - \mathbf{r}_2)^T\ (\Delta\mathbf{r}_1 - \Delta\mathbf{r}_2) = 0 \tag{3.3}$$

If the infinitesimal displacements $\Delta\mathbf{r}_1$ and $\Delta\mathbf{r}_2$ satisfy this equation, so that (3.1) holds, $\Delta\mathbf{r}_1$ and $\Delta\mathbf{r}_2$ are virtual displacements for this example.

A differential displacement of a coordinate **r**, with respect to the generalized coordinates satisfying (3.3) (i.e. a virtual displacement) is

$$\Delta\mathbf{r}_i = \sum_{j=1}^{n} \frac{\delta\mathbf{r}_i}{\delta q_i}\ \Delta q_i, \qquad i = 1, 2, \cdots, k \tag{3.4}$$

where $\Delta q_1, \ldots, \Delta q_n$ are differential displacements of the generalized coordinates. This equation is familiar, if for example, we think about a differential of a function $f(x,y)$, where in our case the function is $\Delta\mathbf{r}_i(q_1, \ldots, q_n)$.

Forces.

Now, suppose that the system is in equilibrium, this means that every particle is in equilibrium, then the total force acting on each particle is zero. This implies that the work done by each set of virtual displacements on a particle is zero. The total work in the system of k particles done by any set of virtual displacements is also zero, that is

$$\sum_{i=1}^{k} \mathbf{F}_i^T \, \Delta\mathbf{r}_i = 0 \tag{3.5}$$

where $\mathbf{F}_i$ is the total force on particle i. This force is the sum of the external force $\mathbf{f}_i$ and the constraint force $\mathbf{f}_i^{(a)}$. We show now, that the constraint forces between a pair of particles, directed along the radial vector connecting the two particles is zero.

Consider once again the example of the two particles with (3.1) as a constraint. The constraint force is exerted by the rigid massless wire, along the radial vector connecting the two particles. The force exerted on particle 1 by the wire is

$$\mathbf{f}_1^{(a)} = c\,(\mathbf{r}_1 - \mathbf{r}_2)$$

where c is a magnitude and $(\mathbf{r}_1 - \mathbf{r}_2)$ indicates the direction of the force. There exists a reaction force on particle 2 exerted by the wire, that is equal in magnitude to $\mathbf{f}_1^{(a)}$, but with an opposite direction (law of action and reaction), that is

$$\mathbf{f}_2^{(a)} = -c\,(\mathbf{r}_1 - \mathbf{r}_2)$$

The work done by the constraint forces with virtual displacements $\Delta\mathbf{r}_1$ and $\Delta\mathbf{r}_2$ is

$$(\mathbf{f}_1^{(a)})^T \,\Delta\mathbf{r}_1 + (\mathbf{f}_2^{(a)})^T \,\Delta\mathbf{r}_2 = c(\mathbf{r}_1 - \mathbf{r}_2)\Delta\mathbf{r}_1 - c(\mathbf{r}_1 - \mathbf{r}_2)\Delta\mathbf{r}_2 = c(\mathbf{r}_1 - \mathbf{r}_2)(\Delta\mathbf{r}_1 - \Delta\mathbf{r}_2)$$

This product is zero, because $\Delta\mathbf{r}_1$ and $\Delta\mathbf{r}_2$ are virtual displacements and therefore satisfy (3.3). Then, there is not any work done by the radial constraint forces on the system. We now assume that all the constraint forces between each pair of links are radial, so that

$$\sum_{i=1}^{k} (\mathbf{f}_i^{(a)})^T \,\Delta\mathbf{r}_i = 0$$

Therefore, the total work in the system done by the $\mathbf{F}_i$'s is

$$\sum_{i=1}^{k} \mathbf{F}_i^T \,\Delta\mathbf{r}_i = \sum_{i=1}^{k} \mathbf{f}_i^T \Delta\mathbf{r}_i + \sum_{i=1}^{k} (\mathbf{f}_i^{(a)})^T \Delta\mathbf{r}_i = \sum_{i=1}^{k} \mathbf{f}_i^T \Delta\mathbf{r}_i = 0 \tag{3.6}$$

This equation does not involve any constraint forces, only known external forces, it expresses the principle of virtual work, which is stated as follows:

Principle of virtual work. The work done by external forces corresponding to any set of virtual displacements is zero.

We can use this principle when the constraint forces do not produce any work. This is the case of rigid bodies in motion, where rigidity is the only constraint. In other words, the distance between any pair of particles in the body must remain constant, while the body is in motion. This is equivalent to an infinite number of constraints of the form (3.1). In some situations like motion in the presence of magnetic fields, the principle of virtual work does not hold. However, in the rest of the analysis we assume that it does.

In (3.6) we cannot say that every $\mathbf{f}_i$ is zero itself, because the virtual displacements $\Delta\mathbf{r}_i$ are not independent. The $\mathbf{f}_i$'s will be zero, if the work is done with virtual displacements of the generalized coordinates. Before we do this, let us consider systems that are not in equilibrium. We now state D'Alambert's principle:

D'Alamberts's principle. In a system that is not in equilibrium, if we introduce a fictitious additional force $-\dot{\mathbf{p}}_i$ on particle i for each i, where $\mathbf{p}_i$ is the momentum of the particle i, then each particle will be in equilibrium.

Now, we replace $\mathbf{F}_i$ by $\mathbf{F}_i - \dot{\mathbf{p}}_i$ in (3.5), that is

$$\sum_{i=1}^{k} (\mathbf{F}_i - \dot{\mathbf{p}}_i)^T \, \Delta\mathbf{r}_i = 0$$

As before, we discard the constraint forces using the principle of virtual work, then

$$\sum_{i=1}^{k} \mathbf{f}_i^T \, \Delta\mathbf{r}_i - \sum_{i=1}^{k} \dot{\mathbf{p}}_i^T \, \Delta\mathbf{r}_i = 0 \tag{3.7}$$

In the following we are going to look for substitutions of both summations for others, in terms of energy. We start with the first summation. Let us express now each $\Delta\mathbf{r}_i$ in terms of the corresponding virtual displacements of generalized coordinates as in (3.4). The virtual work done by the forces $\mathbf{f}_i$ is given by

$$\sum_{i=1}^{k} \mathbf{f}_i^T \Delta\mathbf{r}_i = \sum_{i=1}^{k} \sum_{j=1}^{n} \mathbf{f}_i^T \frac{\delta\mathbf{r}_i}{\delta q_j} \, \Delta q_j = \sum_{j=1}^{n} \Psi_j \, \Delta q_j \tag{3.8}$$

where

$$\Psi_j = \sum_{i=1}^{k} \mathbf{f}_i^T \frac{\delta \mathbf{r}_i}{\delta q_j}$$

is called the *j*th generalized force.

The second summation in (3.7) can be written as

$$\sum_{i=1}^{k} \dot{\mathbf{p}}_i^T \, \Delta \mathbf{r}_i = \sum_{i=1}^{k} m_i \ddot{\mathbf{r}}_i{}^T \Delta \mathbf{r}_i = \sum_{i=1}^{k} \sum_{j=1}^{n} m_i \ddot{\mathbf{r}}^T \frac{\delta \mathbf{r}_i}{\delta q_j} \Delta q_j \tag{3.9}$$

provided that $\mathbf{p}_i = m_i \dot{\mathbf{r}}_i$.

Let us take a look at the following derivative

$$\sum_{i=1}^{k} \frac{d}{dt} \left[m_i \dot{\mathbf{r}}_i{}^T \, \frac{\delta \mathbf{r}_i}{\delta q_j} \right] = \sum_{i=1}^{k} m_i \ddot{\mathbf{r}}_i{}^T \, \frac{\delta \mathbf{r}_i}{\delta q_j} + \sum_{i=1}^{k} m_i \dot{\mathbf{r}}_i{}^T \frac{d}{dt} \left[\frac{\delta \mathbf{r}_i}{\delta q_j} \right]$$

we rewrite this as

$$\sum_{i=1}^{k} m_i \ddot{\mathbf{r}}_i{}^T \frac{\delta \mathbf{r}_i}{\delta q_j} = \sum_{i=1}^{k} \frac{d}{dt} \left[m_i \dot{\mathbf{r}}_i{}^T \, \frac{\delta \mathbf{r}_i}{\delta q_j} \right] - \sum_{i=1}^{k} m_i \dot{\mathbf{r}}^T \frac{d}{dt} \left[\frac{\delta \mathbf{r}_i}{\delta q_j} \right] \tag{3.10}$$

Now, let us differentiate (3.2) with respect to time

$$\dot{\mathbf{r}}_i = \frac{d}{dt} \mathbf{r}_i = \sum_{j=1}^{n} \frac{\delta \mathbf{r}_i}{\delta q_j} \dot{q}_j = \mathbf{v}_i \tag{3.11}$$

where $\mathbf{v}_i$ is the linear velocity of the particle i with respect to the generalized coordinate frame. By differentiating it with respect to $\dot{q}_j$ we get

$$\frac{\delta}{\delta \dot{q}_j} \mathbf{v}_i = \frac{\delta}{\delta q_j} \mathbf{r}_i$$

Let us differentiate the right-hand side with respect to time and let us use (3.11), that is

$$\frac{d}{dt} \left[\frac{\delta \mathbf{r}_i}{\delta q_j} \right] = \frac{\delta \dot{\mathbf{r}}_i}{\delta q_j} = \frac{\delta}{\delta q_j} \sum_{h=1}^{n} \frac{\delta \mathbf{r}_i}{\delta q_h} \dot{q}_h = \sum_{h=1}^{n} \frac{\delta^2 \mathbf{r}_i}{\delta q_j \delta q_h} \dot{q}_h = \frac{\delta \mathbf{v}_i}{\delta q_j}$$

Now, by substituting from the last three equations into (3.10) we get an important expression

$$\sum_{i=1}^{k} m_i \ddot{\mathbf{r}}_i{}^T \frac{\delta \mathbf{r}_i}{\delta q_j} = \sum_{i=1}^{k} \frac{d}{dt} \left[m_i \mathbf{v}_i^T \, \frac{\delta \mathbf{v}_i}{\delta \dot{q}_j} \right] - \sum_{i=1}^{k} m_i \mathbf{v}_i^T \frac{\delta \mathbf{v}_i}{\delta q_j} \tag{3.12}$$

Energies.

We define the kinetic energy of the system as

$$K = \sum_{i=1}^{k} \frac{1}{2} m_i \mathbf{v}_i^T \mathbf{v}_i$$

Let us differentiate K with respect to $\dot{q}_j$

$$\frac{\delta}{\delta \dot{q}_j} K = \sum_{i=1}^{k} \frac{1}{2} m_i \frac{\delta}{\delta \dot{q}_j} \mathbf{v}_i^T \ \mathbf{v}_i + \sum_{i=1}^{k} \frac{1}{2} m_i \mathbf{v}_i^T \frac{\delta}{\delta \dot{q}_j} \mathbf{v}_i = \sum_{i=1}^{k} m_i \mathbf{v}_i^T \frac{\delta}{\delta \dot{q}_j} \mathbf{v}_i$$

Since the inner product is commutative. Also we see that (3.12) can be written in terms of K as

$$\sum_{i=1}^{k} m_i \ddot{\mathbf{r}}_i{}^T \frac{\delta \mathbf{r}_i}{\delta q_j} = \frac{d}{dt} \frac{\delta}{\delta \dot{q}_j} K - \frac{\delta}{\delta q_j} K \tag{3.13}$$

Substitution of (3.13) into (3.9) yields

$$\sum_{i=1}^{k} \dot{p}_i{}^T \Delta \mathbf{r}_i = \sum_{j=1}^{n} \left(\frac{d}{dt} \frac{\delta}{\delta \dot{q}_j} K - \frac{\delta}{\delta q_j} K \right) \Delta q_j \tag{3.14}$$

Finally, we substitute (3.8) and (3.14) into (3.7) to get

$$\sum_{j=1}^{k} \Psi_j \Delta q_j - \sum_{j=1}^{n} \left[\frac{d}{dt} \frac{\delta}{\delta \dot{q}_j} K - \frac{\delta}{\delta q_j} K \right] = 0$$

or

$$\sum_{j=1}^{k} \left\{ \frac{d}{dt} \frac{\delta}{\delta \dot{q}_j} K - \frac{\delta}{\delta q_j} K - \Psi_j \right\} \Delta q_j = 0$$

If we remember that the virtual displacements of the generalized coordinates are linear independent, we conclude that each term in this equation is zero, that is

$$\frac{d}{dt} \frac{\delta}{\delta \dot{q}_j} K - \frac{\delta}{\delta q_j} K = \Psi_j, \qquad j = 1, 2, \cdots, n \tag{3.15}$$

We can now define the generalized force, as the difference of an externally applied generalized force τ_j and another due to a potential field $V(q)$. The value of this force in terms of $V(q)$ is

$$f_{v_j} = \frac{\delta}{\delta q_j}V(q)$$

then

$$\Psi_j = \tau_j - \frac{\delta}{\delta q_j}V(q)$$

$V(q)$ is called the potential energy. Substituting above into (3.15) yields

$$\frac{d}{dt}\frac{\delta}{\delta \dot{q}_j}K - \frac{\delta}{\delta q_j}K = \tau_j - \frac{\delta}{\delta q_j}V$$

or

$$\frac{d}{dt}\frac{\delta}{\delta \dot{q}_j}K - \frac{\delta}{\delta q_j}(K - V) = \tau_j \tag{3.16}$$

Lagrangian.

The Lagrangian of a system is defined to be the difference between the kinetic energy and the potential energy

$$L = K - V \tag{3.17}$$

Notice that

$$\frac{\delta}{\delta \dot{q}_j}L = \frac{\delta}{\delta \dot{q}_j}K$$

because V is a function of q only.

Equation of motion.

We use the last two equations to rewrite (3.16) as follows

$$\frac{d}{dt}\frac{\delta}{\delta \dot{q}_j}L - \frac{\delta}{\delta q_j}L = \tau_j, \qquad j = 1, 2, \cdots, n$$

This equation holds for j=1,2, ... n, where n is the number of generalized coordinates. Thus we can write the equation in vectorial form as

$$\frac{d}{dt}\frac{\delta}{\delta \dot{\mathbf{q}}}L - \frac{\delta}{\delta \mathbf{q}}L = \tau \tag{3.18}$$

where the dimension of τ, q, and $\dot{q}$ is equal to the number of degrees of freedom of the system.

Equation (3.18) is called the Euler–Lagrange equation of motion. It will be used to derive the dynamical model of the general configuration of the robot manipulator.

3.2 KINETIC ENERGY.

The Euler–Lagrange dynamical equation is very useful when the Lagrangian is known. In order to apply this method to the case of robot manipulators, we need to calculate expressions for the kinetic and potential energies. We are going to explore these expressions in the following analysis.

Suppose that we have an object made of a continuum of particles. This object has density of mass σ. Let **B** denote the region of the three–dimensional space occupied by the body. We also use **B** to denote a range of integrals. The total mass of the object is

$$m = \int_{\mathbf{B}} \sigma(x, y, z)\ dxdydz$$

The kinetic energy of the body is given by

$$K = \frac{1}{2}\int_{\mathbf{B}} \mathbf{v}^T(x, y, z)\mathbf{v}(x, y, z)\sigma(x, y, z)\ dxdydz = \frac{1}{2}\int_{\mathbf{B}} \mathbf{v}^T\mathbf{v}\ dm \tag{3.19}$$

where dm denotes the infinitesimal mass of the body in (x,y,z).

When the body is in motion, different parts of it will move at different velocities. Let us consider the center of mass of the object with coordinates (x_c, y_c, z_c), defined by

$$x_c = \frac{1}{m}\int_{\mathbf{B}} x\ dm \qquad y_c = \frac{1}{m}\int_{\mathbf{B}} y\ dm \qquad z_c = \frac{1}{m}\int_{\mathbf{B}} z\ dm$$

Let us express these equations in a more compact form in terms of $\mathbf{r}$, the coordinate vector of a point in the body. We have

$$\mathbf{r}_c = \frac{1}{m}\int_{\mathbf{B}} \mathbf{r}\ dm \tag{3.20}$$

In other words, the vector $\mathbf{r}_c$ given by (3.20), is the position of the center of mass with respect to the reference frame. An alternative representation is

$$\int_{\mathbf{B}} (\mathbf{r}_c - \mathbf{r})\ dm = 0$$

The velocity of a particle in the body with respect to an inertial frame, is given by the sum of the linear velocity of the center of mass, plus the relative linear velocity of the body with respect to the center of mass, that is

$$\mathbf{v} = \mathbf{v}_c + \mathbf{w} x \mathbf{r} \tag{3.21}$$

where $\mathbf{v}_c$ is the linear velocity of the center of mass, **wxr** is the linear velocity of the particle, **r** is the vector from the center of mass to the particle, and **w** is the angular velocity of the particle with respect to the center of mass. We can also express this equation with respect to a moving coordinate frame attached to the center of mass. We can do this by multiplying by a homogeneous transformation ${}^cT_i$, that represents a transformation of coordinates from the frame in the center of mass c to the inertial frame i.

When we are talking about position vectors, that are transformed into a new frame, we consider that these vectors are rotated and translated, but when we differentiate them with respect to time to get velocity vectors, that correspond to changes in position, all the translational terms are eliminated, because they represent constants added to the coordinates. Then, we only consider the rotational effect of the transformations in the analysis of velocity vectors.

In order to refer (3.21) to the moving frame, we multiply it by a rotational matrix R^{-1}, where R is the rotational transformation from the frame in the center of mass, to the inertial frame. As we saw before R is orthogonal, then we have

$$R^T(\mathbf{v}_c + \mathbf{w} x \mathbf{r}) = R^T\mathbf{v}_c + (R^T\mathbf{w}) \text{ x } (R^T\mathbf{r})$$

We notice that when we calculate the kinetic energy, it does not matter in which frame the velocity vectors are referred to, because the magnitude of a vector is not affected by any homogeneous transformation. Let us assume that (3.21) is expressed in terms of the moving frame, and that the cross product **wxr** is expressed by the product of a matrix $S(\mathbf{w})$ and **r**, using the property that for any vector **p**, $S(\mathbf{a})\mathbf{p}=\mathbf{a} x \mathbf{p}$, where $S(\mathbf{a})$ is a skew symmetric matrix (see preliminaries). We can write

$$\mathbf{v} = \mathbf{v}_c + S(\mathbf{w})\mathbf{r}$$

and $S(\mathbf{w})$ is defined by

$$S(\mathbf{w}) = \begin{bmatrix} o & -w_z & w_y \\ w_z & o & -w_x \\ -w_y & w_x & o \end{bmatrix} \tag{3.22}$$

where $\mathbf{w}=[w_x\ w_y\ w_z]^T$, and by definition $S(\mathbf{w})^T=-S(\mathbf{w})$. Substituting $\mathbf{v}$ into (3.19) yields

$$K = \frac{1}{2}\int_B \left(\mathbf{v}_c + S(\mathbf{w})\mathbf{r}\right)^T \left(\mathbf{v}_c + S(\mathbf{w})\mathbf{r}\right)\, dm$$

we expand now this equation by using the property $(AB)^T=B^TA^T$, we have

$$K = \frac{1}{2}\int_B \mathbf{v}_c^T\mathbf{v}_c\, dm + \frac{1}{2}\int_B \mathbf{v}_c^T S(\mathbf{w})\mathbf{r}\, dm + \frac{1}{2}\int_B \mathbf{r}^T S(\mathbf{w})^T\mathbf{v}_c\, dm + \frac{1}{2}\int_B \mathbf{r}^T S(\mathbf{w})^T S(\mathbf{w})\mathbf{r}\, dm \tag{3.23}$$

We are going to take a look at each of these four terms separately. We say that $K=K_1+K_2+K_3+K_4$, where each K_i corresponds to each of the terms of K in (3.23).

In the first term, let us assume that $\mathbf{v}_c$ is independent of the integration variable, therefore we can move it outside of the integral, that is

$$K_1 = \frac{1}{2} m\, \mathbf{v}_c^T\mathbf{v}_c$$

Notice that this expression is the kinetic energy of a particle of mass m, located at the center of mass, and moving with a velocity $\mathbf{v}_c$. This term is called the translational part of the kinetic energy.

The second term is

$$K_2 = \frac{1}{2}\mathbf{v}_c^T\, S(\mathbf{w})\int_B \mathbf{r}\, dm = 0$$

Recall that we are considering that all the vectors in equation (3.21) are referred to the moving frame. By definition of the center of mass, the vector from the reference coordinate frame to the center of mass $\mathbf{r}_c$ is given by (3.20). In this case this vector must be zero, because the reference frame has its origin at the center of mass itself. Then, the integral in (3.24) must be zero, hence $K_2=0$.

Similarly,

$$K_3 = \frac{1}{2}\int_B \mathbf{r}^T dm\ S(\mathbf{w})^T\mathbf{v}_c = 0$$

Let us rewrite the fourth term K_4 using the facts that, for any two matrices A and B, $Tr(AB)=Tr(BA)$, and that for any two vectors $\mathbf{a}$ and $\mathbf{b}$, $\mathbf{a}^T\mathbf{b}=Tr(\mathbf{a}\mathbf{b}^T)$, where Tr stands for the trace of a matrix, and it is equal to the sum of the elements in its diagonal (see preliminaries for details), then we have

$$K_4 = \frac{1}{2}\int_{\mathbf{B}} (S(\mathbf{w})\mathbf{r})^T(S(\mathbf{w})\mathbf{r})\ dm = \frac{1}{2}\int_{\mathbf{B}} Tr\left[(S(\mathbf{w})\mathbf{r})(S(\mathbf{w})\mathbf{r})^T\right] dm$$

$$= \frac{1}{2}\int_{\mathbf{B}} Tr\left[S(\mathbf{w})\mathbf{r}\mathbf{r}^T S(\mathbf{w})^T\right] dm = \frac{1}{2}Tr\left[S(\mathbf{w})\int_{\mathbf{B}} \mathbf{r}\mathbf{r}^T dm\ S(\mathbf{w})^T\right]$$

Let us define the matrix D as

$$D = \int_{\mathbf{B}} \mathbf{r}\mathbf{r}^T dm = \begin{bmatrix} \int x^2 dm & \int xydm & \int xzdm \\ \int xydm & \int y^2 dm & \int yzdm \\ \int xzdm & \int yzdm & \int z^2 dm \end{bmatrix}$$

and substitute into K_4,

$$K_4 = \frac{1}{2}Tr\left[S(\mathbf{w})^T D\ S(\mathbf{w})^T\right]$$

Now, by substituting the value of $S(\mathbf{w})$ given by (3.22) and D into K_4,, we have

$$K_4 = \frac{1}{2}Tr\left\{\begin{bmatrix} o & -w_z & w_y \\ w_z & o & -w_x \\ -w_y & w_x & o \end{bmatrix}\begin{bmatrix} \int x^2 dm & \int xydm & \int xzdm \\ \int xydm & \int y^2 dm & \int yzdm \\ \int xzdm & \int yzdm & \int z^2 dm \end{bmatrix}\begin{bmatrix} o & -w_z & w_y \\ w_z & o & -w_x \\ -w_y & w_x & o \end{bmatrix}\right\}$$

Now, by multiplying, expanding the trace, and grouping, we get

$$K_4 = \frac{1}{2}\begin{bmatrix} w_x & w_y & w_z \end{bmatrix}\begin{bmatrix} \int (y^2+z^2)dm & -\int xydm & -\int xzdm \\ -\int xydm & \int (x^2+z^2)dm & -\int yzdm \\ -\int xzdm & -\int yzdm & \int (x^2+y^2)dm \end{bmatrix}\begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix} = \frac{1}{2}\mathbf{w}^T I\mathbf{w}$$

where I is called the inertia matrix. K_4 is called the rotational part of the kinetic energy.

Now, substituting the values of K_1,K_2,K_3, and K_4 again into (3.23), we get a familiar expression for the kinetic energy

$$K = \frac{1}{2}m\mathbf{v}_c^T\mathbf{v}_c + \frac{1}{2}\mathbf{w}^T I\mathbf{w}$$

The first term of this formula, is the contribution of the linear motion of the body to the kinetic energy, while the second, is the contribution of the angular motion of the body around itself. Hence this equation represents the overall kinetic energy of the body. The two terms represent energy, then it does not matter in which coordinate frame we calculate them. As we mentioned before, the vector $\mathbf{v}_c$ can be affected in a transformation of coordinates, only by the rotational part. Remember that in the above analysis we considered $\mathbf{v}_c$ referred to the center of mass, but since $\mathbf{v}_c{}^T\mathbf{v}_c$ is the square of the magnitude of the vector $\mathbf{v}_c$, it does not matter which frame the vector is expressed. The value of $\mathbf{w}$ and I depend on the coordinate frame to which they are referred, however, we know that the product $\mathbf{w}^T I\mathbf{w}$ remains the same, no matter which coordinate frame the vectors are referred to [4] (pp. 140), since it represents energy. We calculate I with respect to the center of mass, in order to make the development more simple. The angular velocity must be calculated with respect to the same frame.

3.3 POTENTIAL ENERGY.

We consider a rigid body, where the only source of potential energy is gravity. The potential energy of a particle in the body, located at $\mathbf{r}$ from the base of the object and with mass dm, is given by

$$dV = \mathbf{g}^T\mathbf{r}dm$$

where $\mathbf{g}$ is the gravity vector expressed in the base frame. Hence the overall potential energy of the body is given by

$$V = \int_B \mathbf{g}^T\mathbf{r}dm = \mathbf{g}^T\int_B \mathbf{r}dm$$

Recall the definition of the center of mass given in equation (3.20). We then write the potential energy as

$$V = mg^{\mathrm{T}}\mathbf{r}_c$$

In the next section we apply the results of this section and the last one, to develop expressions for the manipulator's energies.

3.4 MANIPULATOR'S ENERGIES.

Kinetic energy.

Let us consider now the manipulator with n links. As we saw before, the linear and angular velocities of any point can be expressed in terms of the joint variables by a Jacobian matrix, and the derivative of the joint variables. We called these joint variables the generalized coordinates of the manipulator. They represent the number of degrees of freedom.

We can express the linear velocity of the center of mass of the link i by

$$\mathbf{v}_{ci} = J_{v_{ci}}(\mathbf{q})\ \dot{\mathbf{q}}$$

where $J_{v_{ci}}$ is the Jacobian matrix corresponding to the ith link. In a similar way, we have the following relation for the angular velocity of the link i

$$\mathbf{w}_{ci} = {}^{i-1}R_i^T\ J_{w_{ci}}(\mathbf{q})\dot{\mathbf{q}}$$

Recall that the rotation matrix is orthogonal, then ${}^{i-1}R_i^T$ represents the inverse of the rotation from the frame i to the frame $i-1$. We need to introduce this rotation matrix because, as we said before, the angular velocity was expressed in terms of the frame $i-1$, and in this case we want the angular velocity of the center of mass of link i, with respect to all the joint displacements, to be represented in its own frame. Remember that the angular velocity of the link i, due to the rotation of link i and represented in the frame i is zero. For the manipulator that we are considering, the overall kinetic energy is equal to the sum of the kinetic energies of each link, that is

$$K = \frac{1}{2}\sum_{i=1}^{n}\left[m_i\mathbf{v}_{ci}^T\mathbf{v}_{ci} + \mathbf{w}_{ci}^T\ I_i\mathbf{w}_{ci}\right]$$

Now, by substituting $\mathbf{v}_{ci}$ and $\mathbf{w}_{ci}$ into K, we obtain the expression for the overall kinetic energy of the manipulator in terms of the joint variables

$$K = \frac{1}{2}\sum_{i=1}^{n}\Big[m_i(J_{v_{ci}}(\mathbf{q})\dot{\mathbf{q}})^T(J_{v_{ci}}(\mathbf{q})\dot{\mathbf{q}}) + \big({}^{i-1}R_i^T\, J_{w_{ci}}(\mathbf{q})\dot{\mathbf{q}}\big)^T I_i\big({}^{i-1}R_i^T\, J_{w_{ci}}(\mathbf{q})\dot{\mathbf{q}}\big)\Big]$$

by eliminating parenthesis

$$K = \frac{1}{2}\sum_{i=1}^{n}\Big[m_i\dot{\mathbf{q}}^T J_{v_{ci}}(\mathbf{q})^T J_{v_{ci}}(\mathbf{q})\dot{\mathbf{q}} + \dot{\mathbf{q}}^T J_{w_{ci}}(\mathbf{q})^T\, {}^{i-1}R_i(\mathbf{q})\; I_i\, {}^{i-1}R_i(\mathbf{q})^T J_{w_{ci}}(\mathbf{q})\dot{\mathbf{q}}\Big]$$

We can take $\dot{\mathbf{q}}^T$ and $\dot{\mathbf{q}}$ out from the summation and get

$$K = \frac{1}{2}\dot{\mathbf{q}}^T \sum_{i=1}^{n}\Big[m_i J_{v_{ci}}(\mathbf{q})^T J_{v_{ci}}(\mathbf{q}) + J_{w_{ci}}(\mathbf{q})^T\, {}^{i-1}R_i(\mathbf{q})\; I_i\, {}^{i-1}R_i(\mathbf{q})^T J_{w_{ci}}(\mathbf{q})\Big]\, \dot{\mathbf{q}}$$

We can see that the summation is only a function of the joint variable vector $\mathbf{q}$, and depends on the kinematic configuration of the manipulator. Let us call $H(\mathbf{q})$ this summation. We rewrite the above as

$$K = \frac{1}{2}\dot{\mathbf{q}}^T H(\mathbf{q})\dot{\mathbf{q}} \tag{3.24}$$

where

$$H(\mathbf{q}) = \sum_{i=1}^{n}\Big[m_i J_{v_{ci}}(\mathbf{q})^T J_{v_{ci}}(\mathbf{q}) + J_{w_{ci}}(\mathbf{q})^T\, {}^{i-1}R_i(\mathbf{q})\; I_i\, {}^{i-1}R_i(\mathbf{q})^T J_{w_{ci}}(\mathbf{q})\Big] \tag{3.25}$$

Notice that the first term in the summation is a symmetric positive definite matrix, because m_i is always positive and the product of the Jacobian matrix by its transpose yields a symmetric square form. Also the second term is a symmetric positive definite matrix, where the inertia matrix I_i is always positive definite, and the overall product is a symmetric square form. Thus, the sum of positive definite matrices leads to a positive definite matrix. Hence, $H(\mathbf{q})$ is a symmetric positive definite matrix. This agrees with the fact that the kinetic energy is positive, unless the system is at rest. It is a convention to call $H(\mathbf{q})$ the *inertia matrix* of the manipulator, or the inertia tensor of the manipulator. Strictly speaking $H(\mathbf{q})$ is a matrix based on individual inertia tensors.

Potential energy

The expression for the potential energy of the manipulator is in general a sum of terms of the form given in section 3.3. Notice that because $\mathbf{r}_{ci}$ is a function of the joint coordinates, the overall potential energy is only a function of $\mathbf{q}$. It is a convention to denote $V(\mathbf{q})$ the overall potential energy of the manipulator, that is

$$V(\mathbf{q}) = \mathbf{g}^T \sum_{i=1}^{n} \mathbf{r}_{ci}\, m_i$$

where $\mathbf{r}_{ci}$ is the coordinate of the center of mass of the link i, and the inner product $\mathbf{g}^T\mathbf{r}_{ci}$, is the projection of the position of the center of mass of the link, into the vertical axis, in other words, its height with respect to the base.

3.5 DYNAMICAL MODEL.

Lagrangian.

Let us find now the Euler–Lagrange equation of motion of a manipulator, for which the kinetic energy is given by (3.24) and the potential energy is $V(\mathbf{q})$. The Lagrangian is given by

$$L = K - V = \frac{1}{2}\dot{\mathbf{q}}^T H(\mathbf{q})\dot{\mathbf{q}} - V(\mathbf{q})$$

Let us write K as a summation, and substitute it into the Lagrangian

$$\begin{aligned} L &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \dot{q}_i h_{ij}(\mathbf{q})\dot{q}_j - V(\mathbf{q}) \\ &= \frac{1}{2}\left(\dot{q}_1 \sum_{j=1}^{n} j_{1j}\dot{q}_j + \dot{q}_2 \sum_{j=1}^{n} h_{2j}\dot{q}_j + \cdots + \dot{q}_n \sum_{j=1}^{n} h_{nj}\dot{q}_j \right) - V(\mathbf{q}) \end{aligned}$$

Now, we have to calculate the terms of the Euler–Lagrange equation of motion given in (3.18). Let us start with the first term. The partial derivative of L with respect to $\dot{q}_k$ is

$$\frac{\delta}{\delta \dot{q}_k} L = \frac{1}{2}\left(\sum_{j=1}^{n} h_{kj}(\mathbf{q})\dot{q}_j + \sum_{j=1}^{n} \dot{q}_j h_{jk}(\mathbf{q}) \right)$$

The first term is for $k=i$ and the second for $j=k$, but since there are n of these terms, we can write it in terms of j from 1 to n, and since $H(\mathbf{q})$ is symmetric, i.e. $h_{ij}=h_{ji}$, we get

$$\frac{\delta}{\delta \dot{q}_k} L = \frac{1}{2}\sum_{j=1}^{n} 2h_{kj}(\mathbf{q})\dot{q}_j = \sum_{j=1}^{n} h_{kj}(\mathbf{q})\dot{q}_j \tag{3.26}$$

In order to visualize this, consider as an example a 3 degrees of freedom manipulator, then

$$L = \frac{1}{2}(\dot{q}_1 h_{11}\dot{q}_{11} + \dot{q}_1 h_{12}\dot{q}_2 + \dot{q}_1 h_{13}\dot{q}_3 + \dot{q}_2 h_{21}\dot{q}_1 + \dot{q}_2 h_{22}\dot{q}_2 + \dot{q}_2 h_{23}\dot{q}_3$$
$$+ \dot{q}_3 h_{31}\dot{q}_1 + \dot{q}_3 h_{32}\dot{q}_2 + \dot{q}_3 h_{33}\dot{q}_3) - V(\mathbf{q})$$

and

$$\begin{aligned}
\frac{\delta}{\delta \dot{q}_1} L &= \frac{1}{2}(2\dot{q}_1 h_{11} + h_{12}\dot{q}_2 + h_{13}\dot{q}_3 + \dot{q}_2 h_{21} + \dot{q}_3 h_{31}) \\
\frac{\delta}{\delta \dot{q}_2} L &= \frac{1}{2}(\dot{q}_1 h_{12} + h_{21}\dot{q}_1 + 2\dot{q}_2 h_{22} + h_{23}\dot{q}_3 + \dot{q}_3 h_{32}) \\
\frac{\delta}{\delta \dot{q}_3} L &= \frac{1}{2}(\dot{q}_1 h_{13} + \dot{q}_2 h_{23} + h_{31}\dot{q}_1 + h_{32}\dot{q}_2 + 2h_{33}\dot{q}_3)
\end{aligned}$$

but because $h_{ij}=h_{ji}$, we have

$$\begin{aligned}
\frac{\delta}{\delta \dot{q}_1} L &= \frac{1}{2}(2\dot{q}_1 h_{11} + 2\dot{q}_2 h_{12} + 2\dot{q}_3 h_{13}) = \sum_{j=1}^{3} h_{1j}\dot{q}_j \\
\frac{\delta}{\delta \dot{q}_2} L &= \frac{1}{2}(2\dot{q}_1 h_{21} + 2\dot{q}_2 h_{22} + 2\dot{q}_3 h_{23}) = \sum_{j=1}^{3} h_{2j}\dot{q}_j \\
\frac{\delta}{\delta \dot{q}_3} L &= \frac{1}{2}(2\dot{q}_1 h_{31} + 2\dot{q}_2 h_{32} + 2\dot{q}_3 h_{33}) = \sum_{j=1}^{3} h_{3j}\dot{q}_j
\end{aligned}$$

Now, let us differentiate (3.26) with respect to time

$$\frac{d}{dt}\frac{\delta}{\delta \dot{q}_k} L = \sum_{j=1}^{n} h_{kj}(\mathbf{q})\ddot{q}_j + \sum_{j=1}^{n} \frac{d}{dt} h_{kj}(\mathbf{q})\dot{q}_j \tag{3.27}$$

Notice that

$$\frac{d}{dt} h_{kj}(\mathbf{q}) = \sum_{i=1}^{n} \frac{\delta}{\delta q_i} h_{kj}\dot{q}_i$$

by the chain rule. Substituting into (3.27)

$$\frac{d}{dt}\frac{\delta}{\delta \dot{q}_k}L = \sum_{j=1}^{n} h_{kj}(\mathbf{q})\ddot{q}_j + \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\delta}{\delta q_i}h_{kj}(\mathbf{q})\dot{q}_i\dot{q}_j$$

where the change in the order of the summations gives the same result.

Now, the second term of the Euler–Lagrange equation, is the partial derivative of L with respect to q_k,

$$\frac{\delta}{\delta q_k}L = \frac{1}{2}\sum_{j=1}^{n}\sum_{j=1}^{n}\frac{\delta}{\delta q_k}h_{ij}(\mathbf{q})\ \dot{q}_i\dot{q}_j - \frac{\delta}{\delta q_k}V(\mathbf{q})$$

We can write now the Euler–Lagrange equation using the last two equations as

$$\sum_{j=1}^{n} h_{kj}(\mathbf{q})\ddot{q}_j + \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{\delta}{\delta q_i}h_{kj}(\mathbf{q}) - \frac{1}{2}\frac{\delta}{\delta q_k}h_{ij}(\mathbf{q})\right)\dot{q}_i\dot{q}_j - \frac{\delta}{\delta q_k}V(\mathbf{q}) = \tau_k, \qquad k = 1, 2, \cdots, n \tag{3.28}$$

Notice that

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\delta}{\delta q_i}h_{kj}(\mathbf{q})\dot{q}_i\dot{q}_j = \sum_{j=1}^{n}\sum_{i=1}^{n}\frac{\delta}{\delta q_i}h_{kj}(\mathbf{q})\dot{q}_i\dot{q}_j = \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\delta}{\delta q_j}h_{ki}(\mathbf{q})\dot{q}_i\dot{q}_j$$

$$= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{\delta}{\delta q_i}h_{kj}(\mathbf{q}) + \frac{\delta}{\delta q_j}h_{ki}(\mathbf{q})\right)\dot{q}_i\dot{q}_j$$

Hence, the second term of (3.28) is

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\frac{\delta}{\delta q_i}h_{kj}(\mathbf{q}) - \frac{1}{2}\frac{\delta}{\delta q_k}h_{ij}(\mathbf{q})\right)\dot{q}_i\dot{q}_j = \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}\left(\frac{\delta}{\delta q_i}h_{kj}(\mathbf{q}) + \frac{\delta}{\delta q_j}h_{ki}(\mathbf{q}) - \frac{\delta}{\delta q_k}h_{ij}(\mathbf{q})\right)\dot{q}_i\dot{q}_j$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}c_{ijk}(\mathbf{q})\ \dot{q}_i\dot{q}_j \tag{3.29}$$

where the elements c_{ijk} are called the Christoffel symbols of the first kind. Note that because the symmetry of $H(\mathbf{q})$, we have that $c_{ijk}=c_{jik}$. Hence, we only have to calculate half of them.

It is a convention to call $g_k(\mathbf{q})$ the partial derivative of the potential energy with respect to $\mathbf{q}_k$, that is

$$g_k(\mathbf{q}) = \frac{\delta}{\delta q_k} V(\mathbf{q}) \tag{3.30}$$

Let us substitute the last two expressions into the Euler–Lagrange equation of motion (3.28), that is

$$\sum_{j=1}^{n} h_{kj}(\mathbf{q})\ddot{q}_j + \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ijk}(\mathbf{q})\ \dot{q}_i\dot{q}_j - g_k(\mathbf{q}) = \tau_k, \qquad k = 1, 2, \cdots, n$$

Now, defining $c_{kj}(\mathbf{q})$ as

$$c_{kj}(\mathbf{q}) = \sum_{i=1}^{n} c_{ijk}(\mathbf{q})\dot{q}_i = \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{1}{2}\left(\frac{\delta}{\delta q_i} h_{kj}(\mathbf{q}) + \frac{\delta}{\delta q_j} h_{ki}(\mathbf{q}) - \frac{\delta}{\delta q_k} h_{ij}(\mathbf{q}) \right) \dot{q}_i \tag{3.31}$$

we can write the n Euler–Lagrange equations, in a vectorial form as

$$H(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \tau \tag{3.32}$$

which is the dynamical model of a rigid robot manipulator with n degrees of freedom. Any configuration of a robot manipulator without regard to its geometry, the number of degrees of freedom, or the kind of joints, will always have a dynamical model of this form.

As we said before $H(\mathbf{q})$ is called the inertia matrix. $C(\mathbf{q}, \dot{\mathbf{q}})$ is called the centrifugal and Coriolis forces matrix, where the terms of the type $\dot{q}_i^2$, correspond to centrifugal forces directed along the radius of rotation of the joint i, and the terms of the type $\dot{q}_i\dot{q}_j$, for $i \neq j$, correspond to Coriolis forces. Remember that when we were talking about rotations of acceleration vectors, we described the acceleration that those forces produce. The vector $\mathbf{g}(\mathbf{q})$ is called the gravity forces vector. This is the model that we are going to control in later chapters.

3.6 PROPERTIES.

Let us mention some properties of the model (3.32). Consider the derivative of $H(\mathbf{q})$ with respect to time. Using the chain rule, we have that the kjth element of $\dot{H}(\mathbf{q})$ is given by

$$\dot{h}_{kj}(\mathbf{q}) = \sum_{i=1}^{n} \frac{\delta}{\delta q_i} h_{kj}(\mathbf{q}) \dot{q}_i$$

Recall that $C(\mathbf{q}, \dot{\mathbf{q}})$ can be written in terms of a matrix that depends only on $\mathbf{q}$, multiplied by the vector $\dot{\mathbf{q}}$, by the definition of the terms c_{kj}, given in (3.31), that is

$$C(\mathbf{q}, \dot{\mathbf{q}}) = C(\mathbf{q})\dot{\mathbf{q}}$$

where $C(\mathbf{q})$ is the matrix of the elements c_{kj}. It is easy to see from this equation, that $C(\mathbf{q},\mathbf{0})=\mathbf{0}$.

Let us subtract the term $2c_{kj}(\mathbf{q}, \dot{\mathbf{q}})$ from $\dot{h}_{kj}(\mathbf{q})$, and see what happens. We have

$$\dot{h}_{kj}(\mathbf{q}) - 2c_{kj}(\mathbf{q}) = \sum_{i=1}^{n} \left\{ \frac{\delta}{\delta q_i} h_{kj}(\mathbf{q}) \dot{q}_i - \left(\frac{\delta}{\delta q_i} h_{kj}(\mathbf{q}) + \frac{\delta}{\delta q_j} h_{ki}(\mathbf{q}) - \frac{\delta}{\delta q_k} h_{ij}(\mathbf{q}) \right) \dot{q}_i \right\}$$

$$= \sum_{i=1}^{n} \left(\frac{\delta}{\delta q_k} h_{ij}(\mathbf{q}) - \frac{\delta}{\delta q_j} h_{ki}(\mathbf{q}) \right) \dot{q}_i$$

Let us do the same the with the jkth element, to get

$$\dot{h}_{jk}(\mathbf{q}) - 2c_{jk}(\mathbf{q}, \dot{\mathbf{q}}) = \sum_{i=1}^{n} \left(\frac{\delta}{\delta q_j} h_{ik}(\mathbf{q}) - \frac{\delta}{\delta q_k} h_{ji}(\mathbf{q}) \right) \dot{q}_i$$

Since $H(\mathbf{q})$ is symmetric, we have that $h_{ki}=h_{ik}$, hence

$$\dot{h}_{kj}(\mathbf{q}) - 2c_{kj}(\mathbf{q}, \dot{\mathbf{q}}) = -\left(\dot{h}_{jk}(\mathbf{q}) - 2c_{jk}(\mathbf{q}, \dot{\mathbf{q}})\right)$$

In other words, the matrix $\dot{H}(\mathbf{q}) - 2C(\mathbf{q}, \dot{\mathbf{q}})$, is skew symmetric. This property leads to the following result

$$\mathbf{x}^T\left(\dot{H}(\mathbf{q}) - 2C(\mathbf{q}, \dot{\mathbf{q}})\right)\mathbf{x} = 0 \qquad (3.33)$$

where $\mathbf{x}$ is any vector [4] (pp. 143). This property will be very useful in the stability analysis of closed-loop robot manipulator control systems.

A very important property of the model (3.32) is that it is linear in the parameters. We are not going to prove this, but the reader is referred to [4] (pp.301). This means that, although the equation of motion is not linear, the parameters of interest such as link masses, moments of inertia, and so forth, appear as coefficients of known functions of the joint variables. If we define each coefficient as a separate parameter, we have a linear realtionship, and we can write the equation as

$$H(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = Y(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})\Theta = \tau \tag{3.34}$$

where $Y(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})$ is an nxr matrix of known functions, and Θ is an r–dimensional vector of parameters. The property is the key feature of robot manipulators that has been used to design adaptive control algorithms.

In $H(\mathbf{q})$, dependence on $\mathbf{q}$, is always in the form of sines and cosines of θ_i, because it depends on the Jacobian, that depends itself in the homogeneous transformation. Since the functions sine and cosine are bounded for any value of $\mathbf{q}$, $H(\mathbf{q})$ is bounded above and below, that is

$$\alpha I \le H(\mathbf{q}) \le \beta I \tag{3.35}$$

for some positive scalars α and β, $\alpha < \beta$, where the relation is in the sense of positive definite matrices [3] (pp.12).

Since $H(\mathbf{q})$ is a square symmetric positive definite matrix, and for manipulators inertia cannot be zero, its inverse exists (see preliminaries).

Now, suppose that we know the structure of the manipulator's model given by (3.32). Consider that the Lagrangian is given by

$$L = \frac{1}{2}\dot{\mathbf{q}}^T K(\mathbf{q})\dot{\mathbf{q}} - P(\mathbf{q})$$

where $K(\mathbf{q})$ is an inertia matrix and $P(\mathbf{q})$ is the potential energy. In fact the first term is the kinetic energy. Let us build the Euler–Lagrange equation of motion, as follows

$$\frac{\delta}{\delta\dot{\mathbf{q}}}L = K(\mathbf{q})\dot{\mathbf{q}}$$

as we stated in (3.26). Also

$$\frac{d}{dt}\frac{\delta}{\delta\dot{\mathbf{q}}}L = \dot{K}(\mathbf{q})\dot{\mathbf{q}} + K(\mathbf{q})\ddot{\mathbf{q}}$$

and

$$\frac{\delta}{\delta\mathbf{q}}L = \frac{1}{2}\dot{\mathbf{q}}^T\left(\frac{\delta}{\delta\mathbf{q}}K(\mathbf{q})\right)\dot{\mathbf{q}} - \frac{\delta}{\delta\mathbf{q}}P(\mathbf{q})$$

Thus, the equation of motion is

$$K(\mathbf{q})\ddot{\mathbf{q}} + \left[\dot{K}(\mathbf{q}) - \frac{1}{2}\dot{\mathbf{q}}^T\left(\frac{\delta}{\delta\mathbf{q}}K(\mathbf{q})\right)\right]\dot{\mathbf{q}} - \frac{\delta}{\delta\mathbf{q}}P(\mathbf{q}) = \tau$$

If we compare this with the model given by (3.32), we can conclude that $H(\mathbf{q})$ is actually the inertia matrix, i.e. $H(\mathbf{q})=K(\mathbf{q})$, [3] (pp. 11). Also

$$C(\mathbf{q},\dot{\mathbf{q}}) = \dot{H}(\mathbf{q}) - \frac{1}{2}\dot{\mathbf{q}}^T\left(\frac{\delta}{\delta\mathbf{q}}H(\mathbf{q})\right)\dot{\mathbf{q}}$$

We can also write $C(\mathbf{q},\dot{\mathbf{q}})$ as [3] (pp. 13)

$$C(\mathbf{q},\dot{\mathbf{q}}) = A(\mathbf{q})[\dot{\mathbf{q}}\dot{\mathbf{q}}] + B(\mathbf{q})[\dot{\mathbf{q}}^2]$$

where $A(\mathbf{q})$ is an $nxn(n-1)/2$ matrix of Coriolis coefficients, $[\dot{\mathbf{q}}\dot{\mathbf{q}}]$ is an $n(n-1)/2x1$ matrix of joint velocity products, given by

$$[\dot{\mathbf{q}}\dot{\mathbf{q}}] = [\dot{q}_1\dot{q}_2\ \dot{q}_1\dot{q}_3\ \cdots\ \dot{q}_{n-1}\dot{q}_n]^T$$

$B(\mathbf{q})$ is an nxn matrix of centrifugal coefficients, and $[\dot{\mathbf{q}}^2]$ is an nx1 vector given by

$$[\dot{\mathbf{q}}^2] = [\dot{q}_1^2\ \dot{q}_2^2\ \cdots\ \dot{q}_n^2]^T$$

About the vector $\mathbf{g}(\mathbf{q})$, we can only say that it has a bound independent of the value of $\mathbf{q}$, since dependence on $\mathbf{q}$ appears only in terms of sine and cosine functions, in the numerators of its elements [3] (pp. 14).

It is also possible to develop a model considering friction, unmodeled dynamics, external disturbances, and flexible links. However, in this work we only consider the model developed with our assumptions. In Chapter 6 some references are mentioned about adaptive controllers, that solve certain kinds of problems of model uncertainties.

BIBLIOGRAPHY.

[1] Asada, H., and Slotine, J., *Robot Analysis and Control*, John Wiley & Sons, New York, 1986.

[2] Craig, J., *Introduction to Robotics Mechanics and Control,* Addison–Wesley Publishing Co., Reading, MA, 1986.

[3] Craig, J., *Adaptive Control of Mechanical Manipulators,* Addison–Wesley Publishing Co., Reading, MA, 1986.

[4] Spong, M., and Vidyasagar, M., *Robot Dynamics and Control,* John Wiley & Sons, New York, 1989.

4. PD PLUS COMPUTED FEEDFORWARD CONTROLLER

In this chapter we present the analysis of two motion controllers for robot manipulators: computed feedforward and PD plus computed feedforward. The latter has shown to have an exelent performance in experiments [1]. However, no analysis was presented. We study existence and uniqueness of equilibrium points, stability and achievement of the control objective for the overall control system.

4.1 INTRODUCTION.

The motivation that originated these kinds of controllers is that, although recent technological advances and reducing costs in the field of digital electronics, have permitted the employment of microprocessor-based equipment, with high speed and powerful computations, in control of robot manipulators, it is of great interest the use of control techniques that include a reduced number of operations to be done on-line. Among these control techniques are computed feedforward and PD plus computed feedforward. The main advantage of these controllers is that all the feedforward terms can be calculated off-line.

In [5] a PD controller with cancellation of gravity was proposed, using a precomputed feedforward gravity compensation. It was shown that the closed-loop system is stable. The PD plus computed feedforward is an extension of this work, in which a precomputed compensation is introduced in the system for the overall dynamics of the robot.

The chapter is organized as follows. In Section 4.2, we present the dynamical model of a robot manipulator with n degrees of freedom (dof), rigid links, and ideal actuators. Using some important properties of the model, we rewrite the dynamical equation in terms of the state vector. Then, the problem formulation is stated in Section 4.3. The analysis of the computed feedforward controller is presented in section 4.4. In Section 4.5, we present the analysis of the PD plus computed feedforward controller. We present simulation results in Section 4.6, for the manipulator of one degree of freedom (dof). We give some conclusions in Section 4.7, and the references are listed at the end of the chapter.

The main result that we found, is that, although the computed feedforward controller seems to be a natural way to control the robot manipulator in closed-loop, it presents multiple equilibrium points, that vary according to the selection of $q_d(t)$, the desired positions vector, and it cannot be satisfactorily used to control a manipulator. We can eliminate this problem by adding a PD. Now, the system is stable, and the control objective is satisfied. We found an explanation for the good performance of this controller presented in [1].

4.2 DYNAMICAL MODEL.

We consider a robot manipulator to be an open kinematic chain, of n dof, with rigid links, ideal actuators, and without friction on the joints. The dynamical model of the manipulator is obtained by the use of the Euler-Lagrange equation of motion, defined in terms of kinetic and potential energies. It has been shown that the model has the following form

$$H(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \tau \tag{4.1}$$

where $\mathbf{q}$ is the nx1 vector of joint variables, τ is the vector of applied joint torques (or forces). $H(\mathbf{q})$ is an nxn matrix called the inertia matrix, defined in terms of the Jacobian, mass, inertia, and geometry of the manipulator. $H(\mathbf{q})$ is a positive definite matrix, therefore its inverse exists. $C(\mathbf{q},\dot{\mathbf{q}})$ is the nxn matrix of centripetal and Coriolis terms, defined in terms of the variations of the inertia properties, with respect to the joint displacements and velocities. $\mathbf{g}(\mathbf{q})$ is the nx1 vector of gravitational forces, defined in terms of the variations of the potential energy with respect to the joint displacements.

Properties.

1) The matrix $\dot{H}(\mathbf{q}) - 2C(\mathbf{q}, \dot{\mathbf{q}})$ is skew symmetric , hence

$$\mathbf{x}^T\left(\dot{H}(\mathbf{q}) - 2C(\mathbf{q}, \dot{\mathbf{q}})\right)\mathbf{x} = 0 \tag{4.2}$$

2) The matrices $H(\mathbf{q})$, $C(\mathbf{q}, \dot{\mathbf{q}})$, and the vector $\mathbf{g}(\mathbf{q})$, are bounded above and below.

4.3 PROBLEM FORMULATION.

The control problem is stated as follows. Given the bounded vectorial functions $\mathbf{q}_d(t)$, $\dot{\mathbf{q}}_d(t)$, and $\ddot{\mathbf{q}}_d(t)$, representing desired positions, velocities, and accelerations of the joint variables, find a vector τ as a function of time, such that the joint positions, velocities, and accelerations of the manipulator follow the given ones with precision. In other words, we want to determine τ, in such a way that

$$\lim_{t \to \infty} \tilde{\mathbf{q}}(t) = \mathbf{0}$$

where $\tilde{\mathbf{q}}(t)$ is the vector of position errors defined as

$$\tilde{\mathbf{q}}(t) = \mathbf{q}_d(t) - \mathbf{q}(t)$$

It suffices to show that $\tilde{\mathbf{q}}(t) \to \mathbf{0}$, to guarantee that $\dot{\mathbf{q}}(t) \to \dot{\mathbf{q}}_d(t)$, and $\ddot{\mathbf{q}}(t) \to \ddot{\mathbf{q}}_d(t)$. So this is our objective.

Usually, those desired values of $\mathbf{q}_d(t)$, $\dot{\mathbf{q}}_d(t)$, and $\ddot{\mathbf{q}}_d(t)$, are provided by a higher stage called the trajectory generation system. We do not expect that the controller will perform derivatives, therefore we need the three values to be provided.

4.4 COMPUTED FEEDFORWARD CONTROLLER.

This is one of the most elementary model-based control strategies, that can be used. We have to consider that the model of the robot is perfectly known, that is, the matrices $H(\mathbf{q})$,

$C(\mathbf{q},\dot{\mathbf{q}})$, and the vector $\mathbf{g}(\mathbf{q})$. We are interested in this kind of controller, because once $\mathbf{q}_d(t)$, $\dot{\mathbf{q}}_d(t)$, and $\ddot{\mathbf{q}}_d(t)$ are known, the implementation is fairly straight forward. We can calculate the control action τ from (1), just by substituting these values. In other words, we calculate $H(\mathbf{q}_d)$, $C(\mathbf{q}_d,\dot{\mathbf{q}}_d)$, and the vector $\mathbf{g}(\mathbf{q}_d)$ offline. Then, we can generate the values of τ to be applied.

Notice that this is an open-loop controller, and in consequence we have all the disadvantages of an open-loop control system. The block diagram is shown in Figure 1 on page 68.

The behaviour of the system is obtained by replacing τ from (4.1), with its precomputed value. Let us express this open-loop equation in terms of the state vector $[\tilde{\mathbf{q}}\ \dot{\tilde{\mathbf{q}}}]^T$, as follows

$$\frac{d}{dt}\begin{bmatrix}\tilde{\mathbf{q}}(t)\\ \dot{\tilde{\mathbf{q}}}(t)\end{bmatrix} = \begin{bmatrix}\dot{\tilde{\mathbf{q}}}(t)\\ H(\mathbf{q})^{-1}\left\{(H(\mathbf{q})-H(\mathbf{q}_d))\ddot{\mathbf{q}}_d + (C(\mathbf{q},\dot{\mathbf{q}})-C(\mathbf{q}_d,\dot{\mathbf{q}}_d))\dot{\mathbf{q}}_d - C(\mathbf{q},\dot{\mathbf{q}})\dot{\tilde{\mathbf{q}}} + \mathbf{g}(\mathbf{q}) - \mathbf{g}(\mathbf{q}_d)\right\}\end{bmatrix} \tag{4.3}$$

where $\mathbf{q} = \mathbf{q}_d - \tilde{\mathbf{q}}$. This equation represents an ordinary nonlinear nonautonomous differential equation. It is clear that the origin $[\tilde{\mathbf{q}}\ \dot{\tilde{\mathbf{q}}}]^T = \mathbf{0}$ is an equilibrium. By definition of equilibrium, if $\mathbf{q}(0) = \mathbf{q}_d(0)$, and $\dot{\mathbf{q}}(0) = \dot{\mathbf{q}}_d(0)$, then $\tilde{\mathbf{q}}(t) = \mathbf{0}$, for all $t \geq 0$.

However, this equilibrium is in general not unique, and it is difficult to guarantee that the initial conditions are identically equal to the desired initial conditions. Then, what can we say about the behaviour of $\mathbf{q}(t)$ and $\dot{\mathbf{q}}(t)$? The answer to this question could come from the stability analysis of the equilibrium points. It is possible that the system has multiple equilibrium points, then we can discard the option of global asymptotic stability. So we can only expect local asymptotic or exponential stability properties. In order to determine these, we will need more specific information about a particular configuration of a manipulator, and with different configurations we can have different behaviours.

For simplicity, we present an example of the computed feedforward controller with a one degree of freedom manipulator, in other words, a pendulum. This kind of manipulator is extensively used in experiments for control, and particularly in robotics.

Feedforward control of a pendulum.

Consider an ideal pendulum with length l, mass m concentrated in its extreme, exposed to the gravity force g. We can use the Euler–Lagrange equation of motion to represent its dynamical model. This model is given by

$$ml^2\ddot{q} + mgl\sin(q) = \tau \tag{4.4}$$

which has the form (1), with

$$H(q) = ml^2, \quad C(q,\dot{q}) = 0, \quad g(q) = mgl\sin(q)$$

The open–loop state equation is

$$\frac{d}{dt}\begin{bmatrix}\tilde{q}(t)\\ \dot{\tilde{q}}(t)\end{bmatrix} = \begin{bmatrix}\dot{\tilde{q}}(t)\\ \frac{1}{ml^2}(mgl\sin(q) - mgl\sin(q_d))\end{bmatrix} \tag{4.5}$$

Clearly, the origin is an equilibrium, and so is $[\tilde{q}\ \dot{\tilde{q}}]^T = [2n\pi\ 0]^T$, for $n \in \mathbf{Z}$.

Consider the particular case in which $q_d(t)$=0. Now, the state space representation, is the equation of a free pendulum, given by

$$\frac{d}{dt}\begin{bmatrix}\tilde{q}(t)\\ \dot{\tilde{q}}(t)\end{bmatrix} = \begin{bmatrix}\dot{\tilde{q}}(t)\\ -\frac{g}{l}\sin(\tilde{q})\end{bmatrix} \tag{4.6}$$

Notice that the equilibrium points are now $[\tilde{q}\ \dot{\tilde{q}}]^T = [n\pi\ 0]^T$, which means that we have more equilibriums. It is easy to show that the origin is a stable equilibrium, but it is not asymptotically stable, and the equilibriums corresponding to n odd are unstable [3].

With this example, we show that the feedforward controller cannot satisfy even position objectives. For this reason, this open–loop controller is not satisfactorily used in practice. However, it may be applicable to some special cases, but so far we do not have a general result to prove that it will always work.

We state that despite of its well reasoning foundation and ease of implementation, it is not successfully applicable to the control problem of robot manipulators.

From the practical point of view, it is of great interest to include as less a reduced number of operations in real–time, when implementing a controller, this motivates the inclusion of the precomputed terms. In [6] a PD controller plus a gravity term was suggested. In [5] a modification was made to it in order to include precomputed terms. The PD controller plus computed feedforward is a generalization of this controller, that has shown [1] to perform well. We now discuss its performance.

4.5 PD PLUS COMPUTED FEEDFORWARD CONTROLLER.

The feedforward controller is modified by the addition of a proportional–derivative term (PD), as follows

$$\tau = H(\mathbf{q}_d)\ddot{\mathbf{q}}_d + C(\mathbf{q}_d, \dot{\mathbf{q}}_d)\dot{\mathbf{q}}_d + \mathbf{g}(\mathbf{q}_d) + K_p\tilde{\mathbf{q}} + K_v\dot{\tilde{\mathbf{q}}} \qquad (4.7)$$

where $K_p = K_p^T > 0$, and $K_v = K_v^T > 0$, are nxn positive definite matrices of position and velocity gains. The proportional–derivative term closes the loop, providing an explicit feedback of $\mathbf{q}$ and $\dot{\mathbf{q}}$. In Figure 2 the block diagram of this controller is presented. This is a generalization of the PD controller plus precomputed gravity compensation presented in [5], since it includes feedforward compensation for the overall dynamics of the robot. The experimental results presented in [1] have shown that this controller has an excellent performance, comparable to the quite more complex computed torque controller. These results are surprising because of the relative easiness of the control law. Unfortunately, the stability analysis is not presented, and to the knowledge of the author, it has not been reported before. In the following analysis, we conclude that the good performance obtained, is due to a high proportional gain K_p in the closed–loop.

The closed–loop behaviour is obtained by replacing τ in the equation of the robot (4.1) with the controller. Let us express it in terms of the state vector, that is

$$\frac{d}{dt}\begin{bmatrix}\tilde{\mathbf{q}}(t)\\ \dot{\tilde{\mathbf{q}}}(t)\end{bmatrix} = \begin{bmatrix}\dot{\tilde{\mathbf{q}}}(t)\\ H^{-1}\{(H-H_d)\ddot{\mathbf{q}}_d + (C-C_d)\dot{\mathbf{q}}_d - C\dot{\tilde{\mathbf{q}}} + \mathbf{g} - \mathbf{g}_d - K_p\tilde{\mathbf{q}} - K_v\dot{\tilde{\mathbf{q}}}\}\end{bmatrix} \qquad (4.8)$$

where we have removed the arguments, in order to abreviate notation (H_d means $H(\mathbf{q}_d)$, and so on). Notice that the origin is an equilibrium. Using the same reasoning as [5], we propose the following function

$$V(t,\tilde{q},\dot{\tilde{q}}) = \frac{1}{2}\left(\dot{\tilde{q}}^T H\dot{\tilde{q}} + \tilde{q}^T Q\tilde{q}\right) + \tilde{q}^T g_d + \int_0^t \left[\dot{\tilde{q}}^T\{(H_d - H)\ddot{q}_d + (C_d - C)\dot{q}_d - g\} - \tilde{q}^T \dot{g}_d\right]ds \qquad (4.9)$$

where $Q=Q^T=K_p>\mathbf{0}$. It is easy to check that this function vanishes at the equilibrium, and it is positive otherwise , as long as the entries on Q are large enough. $V(t,\tilde{q},\dot{\tilde{q}})$ dominates the function

$$W_1(\tilde{q},\dot{\tilde{q}}) = \frac{1}{2}\left(\dot{\tilde{q}}^T H\dot{\tilde{q}} + \alpha_1\tilde{q}^T Q\tilde{q}\right)$$

where $\alpha_1 > 0$ is a small enough constant. Notice that $W_1(\tilde{q},\dot{\tilde{q}})$ is a positive definite function. $V(t,\tilde{q},\dot{\tilde{q}})$ is also dominated by the function

$$W_2(\tilde{q},\dot{\tilde{q}}) = \frac{1}{2}\left(\dot{\tilde{q}}^T H\dot{\tilde{q}} + \alpha_2\tilde{q}^T Q\tilde{q}\right)$$

for $\alpha_2 > 0$ is a large enough constant. $W_2(\tilde{q},\dot{\tilde{q}})$ is also a positive definite function. Hence $V(t,\tilde{q},\dot{\tilde{q}})$ is a positive definite function (see preliminaries), it is decrescent, and we propose it as a Lyapunov candidate function.

Differentiating V with respect to time, evaluating along the state trajectories, and using the property that $\dot{H}(q) - 2C(q,\dot{q})$ is skew symmetric, we get

$$\dot{V}(t,\tilde{q},\dot{\tilde{q}}) = -\dot{\tilde{q}}^T K_v\dot{\tilde{q}} \le 0 \qquad (4.10)$$

This implies that the equilibrium is stable in the sense of Lyapunov, and $V(t,\tilde{q},\dot{\tilde{q}})$ is a Lyapunov function. We use the following two lemmas to prove that the control objective is satisfied.

Lemma 4.1 [9] (pp. 232 fact 4)

Let $\mathbf{f}:\mathbf{R}_+ \to \mathbf{R}^n$. If $\mathbf{f} \in L_2^n$ and $\dot{\mathbf{f}} \in L_\infty^n$, then $\mathbf{f}(t) \to \mathbf{0}$ as $t \to \infty$.

Lemma 4.2 [4] (lemma 2.2)

Consider the continuous and differentiable functions $\mathbf{x}:\mathbf{R}_+ \to \mathbf{R}^m$, and $f:\mathbf{R}_+ \to \mathbf{R}_+$. Define the function $V:\mathbf{R}^{m+1} \to \mathbf{R}_+$, given by

$$V(t,\mathbf{x},f) = \mathbf{x}(t)^T K_1\mathbf{x}(t) + f(t) \ge 0$$

where $K_1 \in \mathbf{R}^{mxm}$ is a symmetric positive definite matrix. If there exists a function $\mathbf{z}:\mathbf{R}_+ \to \mathbf{U}$, where **U** is a subspace of $\mathbf{R}^m$, of dimension $p(\leq m)$, such that the derivative of V with respect to time satisfies

$$\dot{V}(t,\mathbf{x},f) = -\mathbf{z}(t)^T K_2 \mathbf{z}(t) \leq 0$$

where $K_2 = K_2^T > \mathbf{0}$, then

$$a) \quad \mathbf{x} \in L_\infty^m$$
$$b) \quad f \in L_\infty$$
$$c) \quad \mathbf{z} \in L_2^p$$

These lemmas imply that $\tilde{\mathbf{q}}, \dot{\tilde{\mathbf{q}}} \in L_\infty^n$, $\dot{\tilde{\mathbf{q}}} \in L_2^n$, and

$$\lim_{t \to \infty} \dot{\tilde{\mathbf{q}}}(t) = \mathbf{0}$$

where $\mathbf{x} = [\tilde{q}\ \dot{\tilde{q}}]^T$,

$$K_1 = \frac{1}{2}\begin{bmatrix} Q & \mathbf{0} \\ \mathbf{0} & H \end{bmatrix}, \qquad f(t) = \int_o^t \mathbf{x}^T \begin{bmatrix} \dot{\mathbf{g}}_d & \mathbf{0} & \mathbf{0} \\ \mathbf{g} & C_d - C & H_d - H \end{bmatrix} \begin{bmatrix} 1 \\ \dot{\mathbf{q}}_d \\ \ddot{\mathbf{q}}_d \end{bmatrix} + \tilde{\mathbf{q}}^T \mathbf{g}_d \, ds$$

and K_2=K_v.

We claim that a large enough K_p=Q, always exists since all the quantities are bounded and it is reasonable to think that $\mathbf{q}_d(t)$, $\dot{\mathbf{q}}_d(t)$, and $\ddot{\mathbf{q}}_d(t)$ are bounded.

We are not able to conclude that the equilibrium is asymptotically stable by the use of LaSalles's theorem, as suggested in [5], because V is nonautonomous. In [6], it is shown that the Matrosov's theorem could be used to guarantee the global asymptotic stability of the origin.

We show that the control objective is satisfied by showing that $\tilde{\mathbf{q}}(t)$ converges to a constant, and this constant is the zero vector. Consider $\dot{\tilde{\mathbf{q}}}(t) = \mathbf{0}$, if $\tilde{\mathbf{q}}(t)$ is a constant, it must be a solution of the closed loop system. We calculated the solution by an iterative method, and it resulted to be zero, for different functions $\mathbf{q}_d(t)$. We conjecture that, in general, the equilibrium is unique when the desired position vector is not a constant, and it does not converge to a constant.

We present the following example, for the manipulator of one dof, in order to illustrate our conjecture.

PD plus feedforward control of a pendulum.

Consider again the model of the ideal pendulum. The closed-loop system is given by

$$\frac{d}{dt}\begin{bmatrix}\tilde{q}(t)\\ \dot{\tilde{q}}(t)\end{bmatrix} = \begin{bmatrix}\dot{\tilde{q}}(t)\\ \frac{1}{ml^2}\left(mgl\sin(q_d-\tilde{q}) - mgl\sin(q_d) - K_p\tilde{q} - K_v\dot{\tilde{q}}\right)\end{bmatrix} \tag{4.11}$$

Clearly, the origin is an equilibrium. If $q_d(t)$ is a constant, we could have additional equilibriums $[\tilde{q}\ \dot{\tilde{q}}]^T = [s\ 0]^T$, where s is the solution of

$$K_p s + mgl(\sin(q_d) - \sin(q_d - \tilde{q})) = 0 \tag{4.12}$$

It is interesting to observe the upper bound of $|s|$, that is

$$|s| \leq \left| -\frac{mgl}{K_p}(\sin(q_d) - \sin(q_d-\tilde{q}))\right| \leq \frac{2mgl}{K_p} \tag{4.13}$$

Notice that if $K_p \to \infty$, then $|s| \to 0$. Consider now the following Lyapunov candidate function

$$V(t,\tilde{q},\dot{\tilde{q}}) = \frac{1}{2}\left(ml^2\dot{\tilde{q}}^2 + K_p\tilde{q}^2\right) + mgl\tilde{q}\sin(q_d) + \int_0^T mgl\dot{\tilde{q}}\sin(\tilde{q}-q_d) - mgl\tilde{q}\dot{q}_d\cos(q_d)\ ds \tag{4.14}$$

which is positive definited for K_p large enough, since it is dominated by the positive definite function

$$W_1(\tilde{q},\dot{\tilde{q}}) = \frac{1}{2}\left(ml^2\dot{\tilde{q}}^2 + \alpha_1 K_p\tilde{q}^2\right)$$

and it dominates the positive definite function

$$W_2(\tilde{q},\dot{\tilde{q}}) = \frac{1}{2}\left(ml^2\dot{\tilde{q}}^2 + \alpha_2 K_p\tilde{q}^2\right)$$

where $\alpha_1, \alpha_2 > 0$, are a large enough constant, and a small enough constant respectively. This also indicates that $V(t,\tilde{q},\dot{\tilde{q}})$ is a decrescent function. Its derivative with respect to time is given by

$$\dot{V}(t,\tilde{q},\dot{\tilde{q}}) = -K_v\dot{\tilde{q}}^2 \leq 0 \tag{4.15}$$

Hence, the origin is stable, and $V(t,\tilde{q},\dot{\tilde{q}})$ is a Lyapunov function. It follows from lemmas 4.1 and 4.2, that

$$\lim_{t \to \infty} \dot{\tilde{q}}(t) = 0$$

Considering this limit, and the state equation, we claim that the next equationhas a unique solution $\tilde{q} = 0$

$$K_p\tilde{q} + mgl(\sin(q_d) - \sin(q_d - \tilde{q})) = 0 \tag{4.16}$$

We used an iterative method to find a solution of this equation, for different functions $q_d(t)$ that does not converge to a constant, and different large enough values of K_p. We found that in all cases $\tilde{q} \to 0$ as $t \to \infty$, satisfying the control objective.

4.6 SIMULATION RESULTS.

We present simulations of the PD controller plus computed feedforward with a pendulum . The parameters of the pendulum were m=1, l=1, and g=10.

In Figure 3, we considered $q_d(t) = \pi/2$. The design parameters were K_p=1/10, and K_v=1. The initial conditions were $q(0) = -3.43$, and $\dot{q}(0) = 0$. In the graph it is shown that $q(t)$ does not converge to $q_d(t)$, but it does to 69.44. Notice that $[\tilde{q}\ \dot{\tilde{q}}]^T = [-67.87\ \ 0]^T$, is an equilibrium and that $|s| = 67.87 \le 2mgl/K_p = 200$. Notice that, although the system started near the origin, and it past it, it did not converge to it. Also notice that the structure of the controller is identical to that proposed in [5]. We conclude that this value of K_p is not large enough.

In Figure 4, we considered the same situation, except that we set K_p=10. Now that K_p has been increased, the control objective is satisfied. It can be seen that the origin is the equilibrium.

In Figure 5, the same values for the constants and initial conditions were used, except that this time the input is $q_d(t)=sin(t)$. Again, as in figure 3, $q(t)$ does not converge to $q_d(t)$, but the position error does to a value around 6.3. Although this value is not a constant, the system regulates.

In Figure 6, we finally increased the value of the proportional gain. We used K_p=5, which is large enough for the position and velocity errors to converge to the origin.

From these results we conclude that a reason for the good performance of this controller presented in [1], is the selection of a large enough K_p.

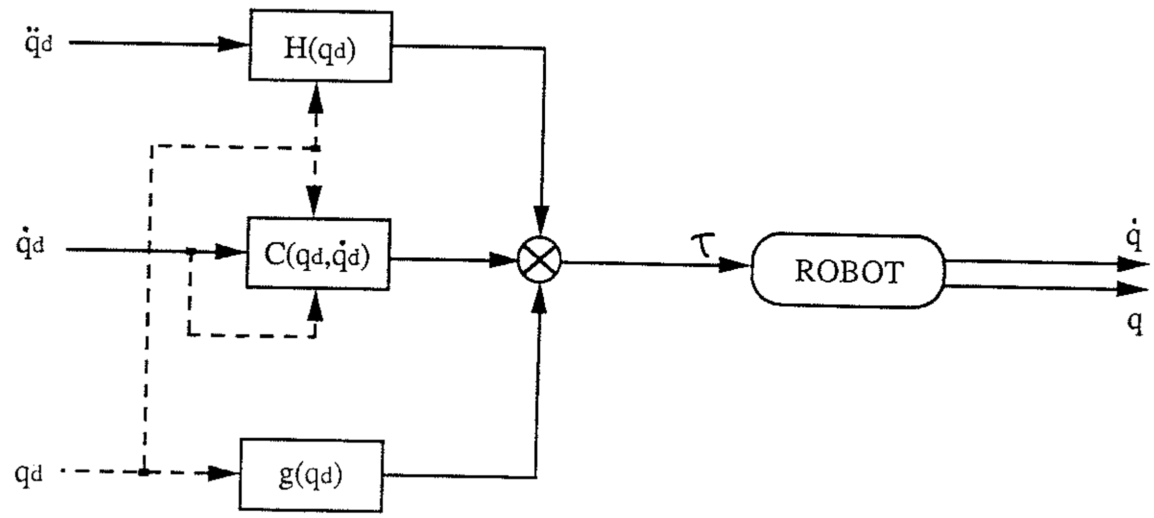


Figure 1. Feedforward Controller.

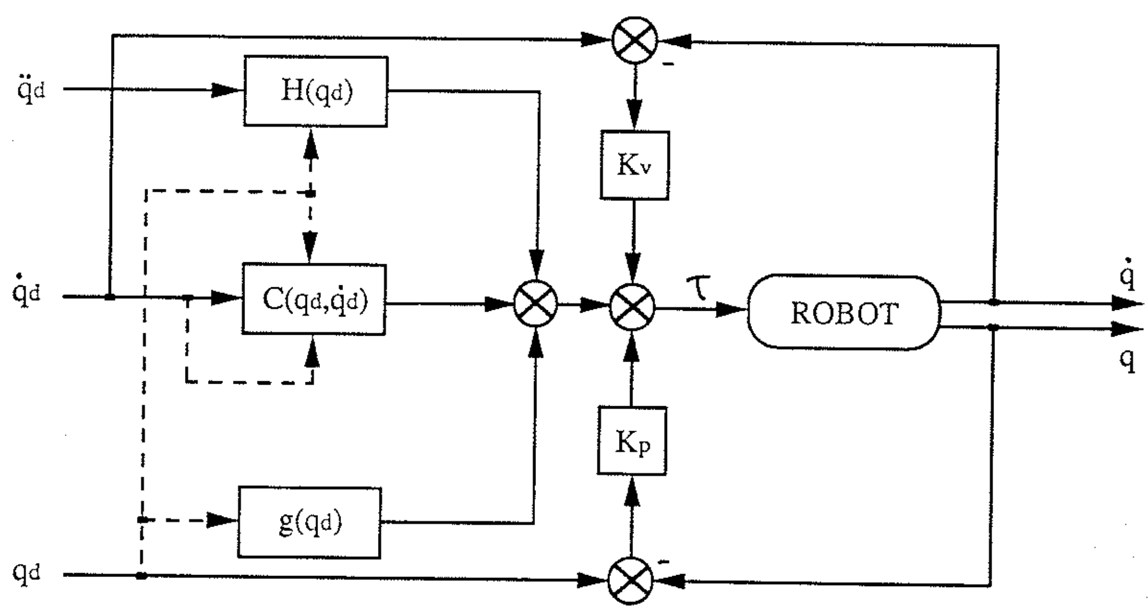


Figure 2. PD plus Feedforward Controller.

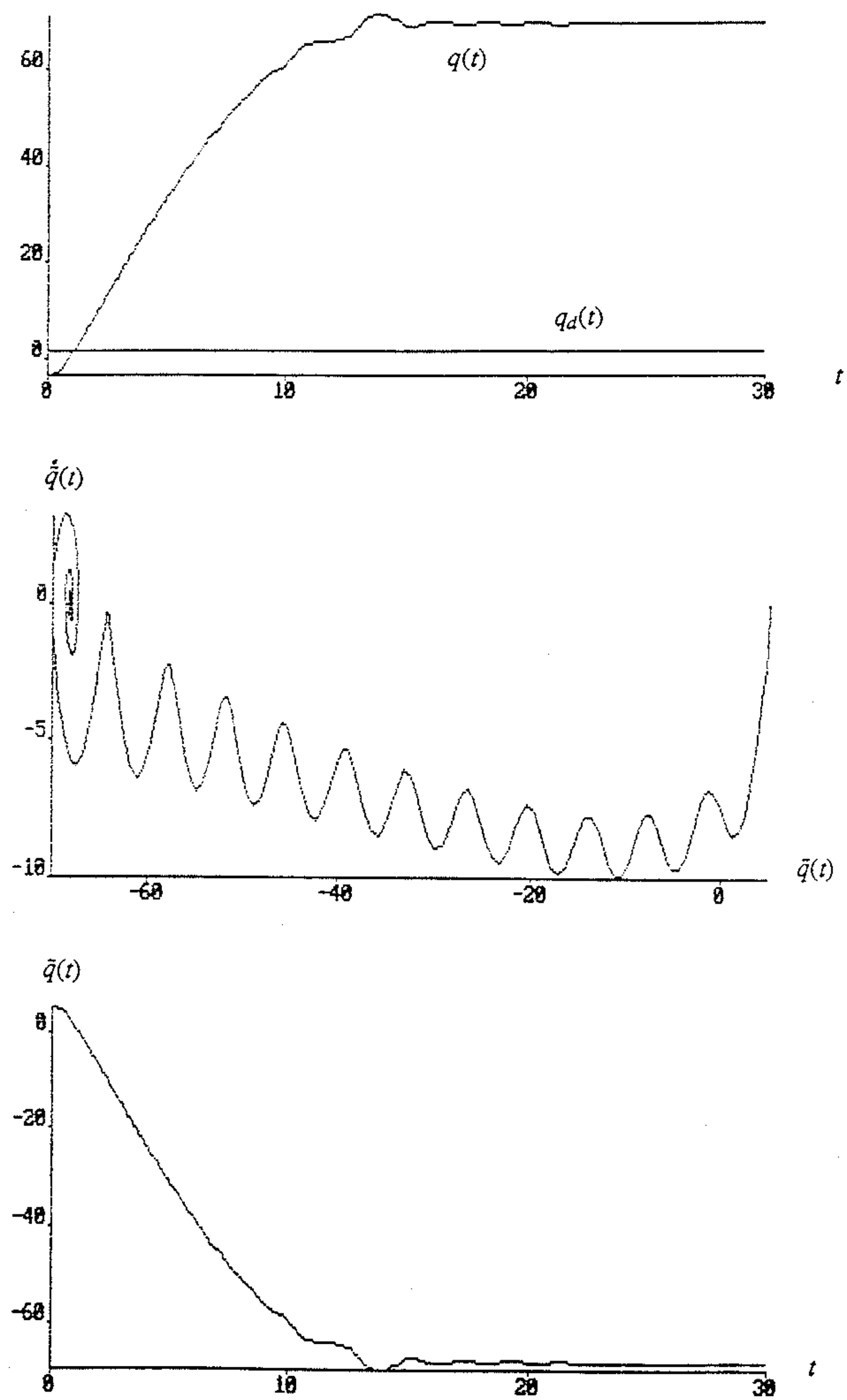


Figure 3. Simulation: $q_d(t) = \pi/2$, Kp=0.1.

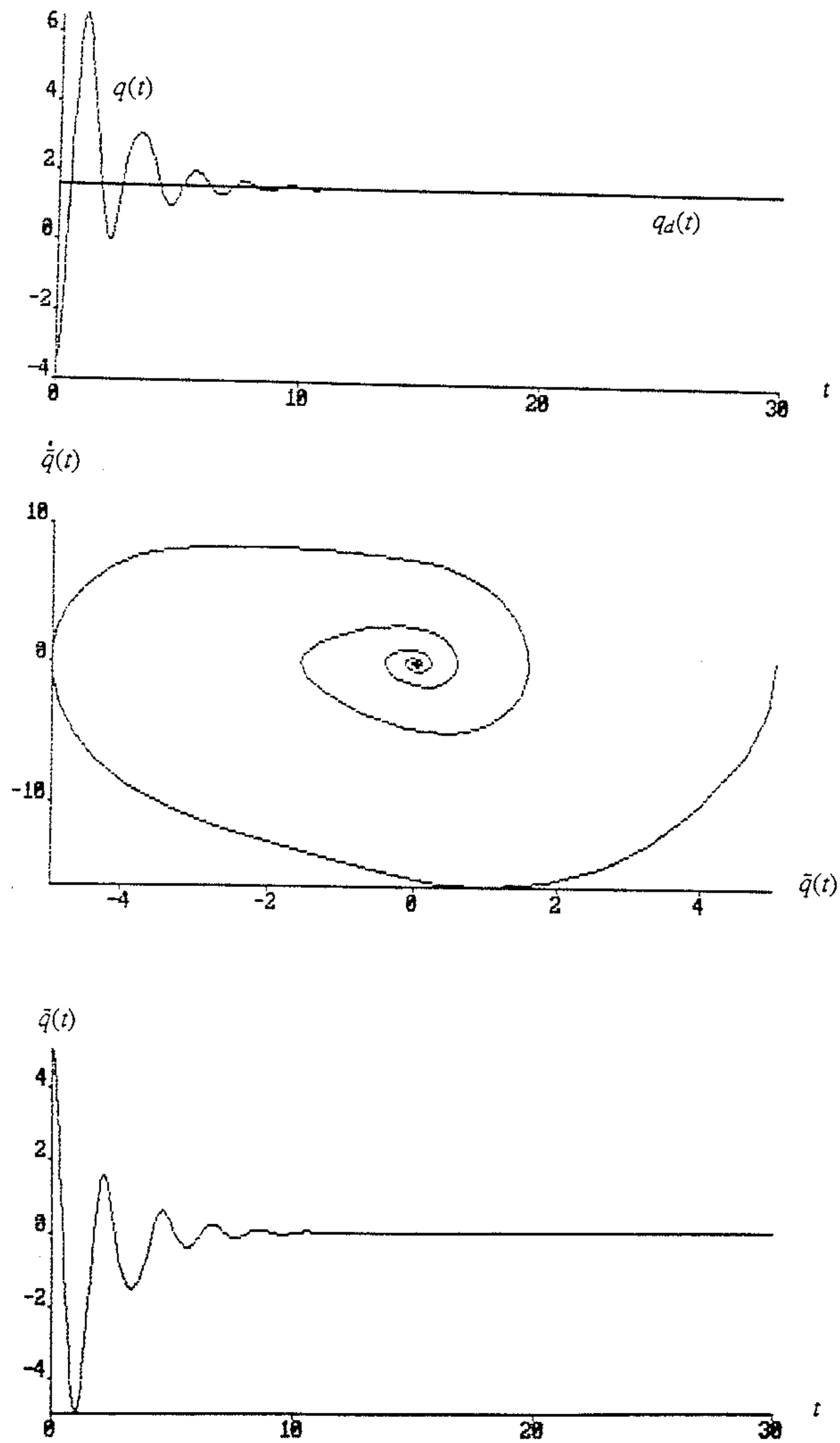


Figure 4. Simulation: $q_d(t) = \pi/2$, Kp=10.

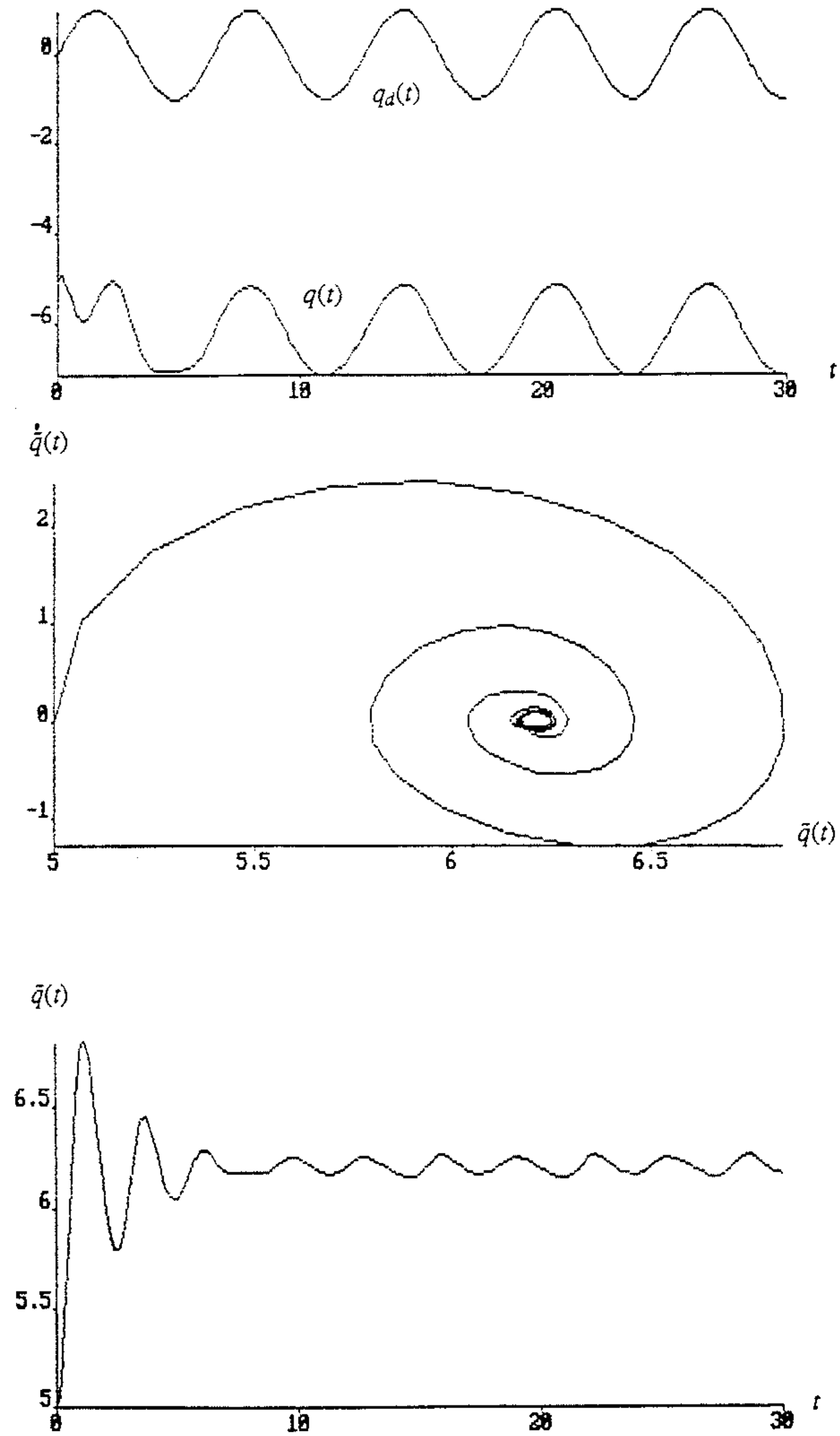


Figure 5. Simulation: $q_d(t) = \sin(t)$, K_p=0.1.

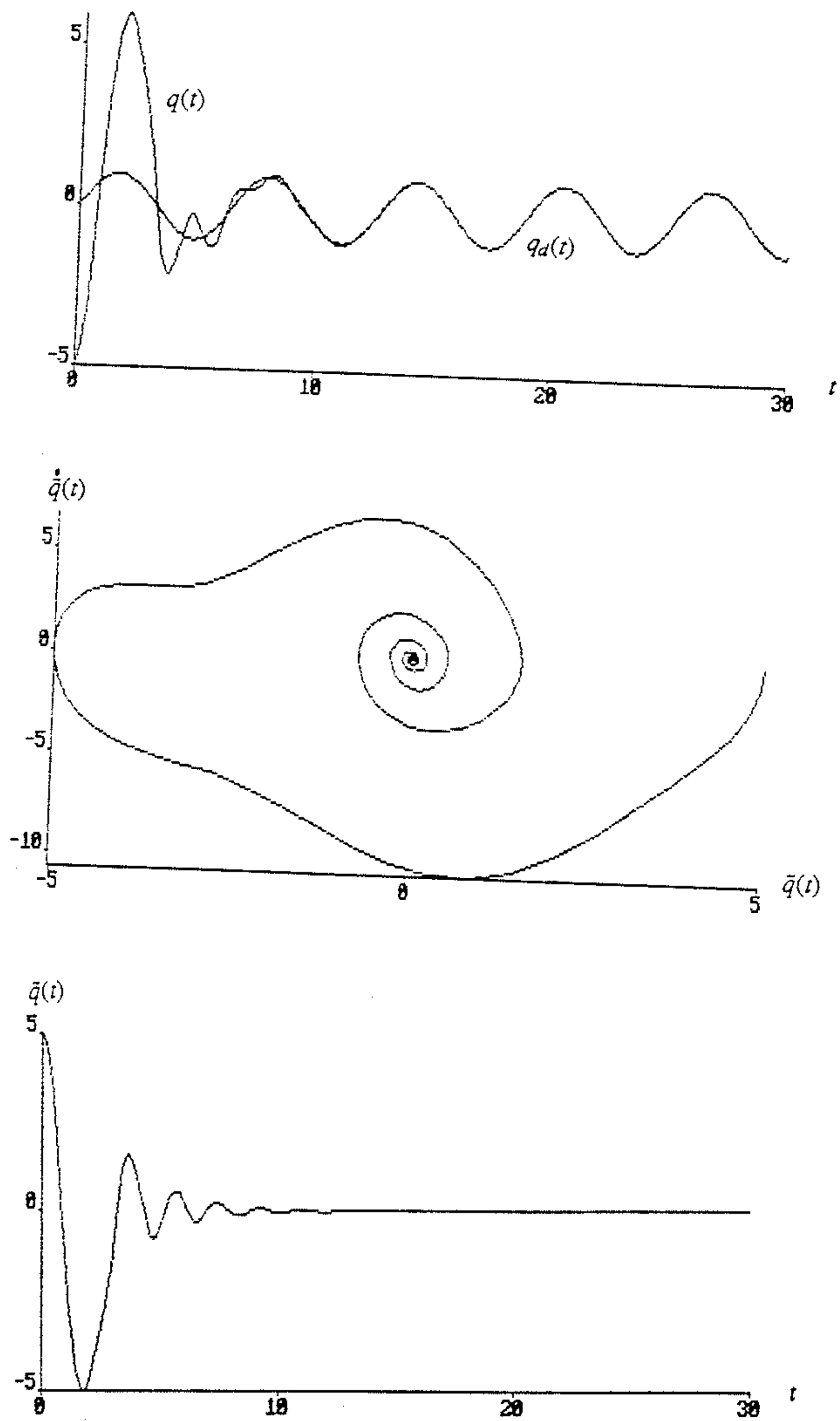


Figure 6. Simulation: $q_d(t) = \sin(t)$, $Kp=5$.

4.7 CONCLUSIONS.

In this work we studied the computed feedforward and the PD plus computed feedforward controllers for robot manipulators. We found that the computed feedforward controller is not convenient for the control of manipulators, since it is not capable to guarantee even a pure position objective.

From the analysis of the PD plus computed feedforward, we conjecture that, the closed–loop system has a unique equilibrium when the vector of desired positions is not a constant, and we show that the origin of the state space of the closed–loop system is stable for large enough values of K_p. This selection guarantees convergence of the velocity and position errors. However, it is still necessary to find a lower bound for K_p.

REFERENCES.

[1] An, H., Atkeson, C., Hollerbach, J., *Model-Based Control of a Manipulator*, The MIT press, , 1988.

[2] Desoer, C., Vidyasagar, M., *Feedback Systems: Input-Output Properties*, Academic Press, New York, 1975.

[3] Kelly, R., *Control de Movimiento de Robots Manipuladores*, DEPFI–UNAM, D–79, Mexico City, 1989.

[4] Kelly, R., Rueda, A., Alcorta, E., "Motion Controller for Robot Manipulators: Computed Feedforward and PD plus Computed Feedforward", Proceedings of the 3rd International Conference on Advances on Comunications and Control, Victoria, Canada, October, 1991, pp. (to be anounced).

[5] Khorrami, F., Özgüner, U., "Decentralized Control for Robot Manipulators via State and Proportional–Integral Feedback", Procc. of the IEEE International Conference on Robotics and Automation, April 24–28, 1988, Philadelphia, PA, vol. 2, pp. 1198–1203.

[6] Paden, B., Panja, R., "Globally Asymptotically Stable PD+ Controller for Robot Manipulators", Int. Journal of Control, vol. 47, No. 6, 1988, pp. 1697–1712.

[7] Vidyasagar, M., *Nonlinear Systems Analysis*, Prentice Hall, New Jersey, 1978.

5. PD PLUS ADAPTIVE FEEDFORWARD COMPENSATION CONTROLLER

In this document we present the design of an adaptive controller for direct current motors, with parametric uncertainties, that can be applied to robot manipulators. We show stability and achievement of the control objective for the overall control system.

5.1 INTRODUCTION.

Electric motors are one of the most used energy source in low power industrial applications. In this chapter, we present the analysis of a controller for direct current motors (DC motors), to be used as actuators in robot manipulator joints, where we need great precision in the following of trajectory specifications. Brushless motors that emulate via a power electronic interface, the linear transfer function of a DC motor, are widely applied in industrial robots. We are interested in those cases in robotics, in which independent joint control is enough to satisfy control objectives.

There exists a great variety of commercial electronic equipment for DC motors, designated to the applications of velocity regulation, and in less scale for position regulation. This is due to the fact that in the majority of the industrial applications, DC motors are used in tasks that involve motion at constant velocity. However, there are applications in which velocity regulation or position regulation is not enough to satisfy certain specifications of motion. Applications in robotics are a typical example, where it is required for the manipulator to follow time-varying trajectories of position and velocity. Electronic equipment for these kinds of applications is practically absent in the market, due to the lack of economic feasability, because of the limited number of applications, but not as a result of the lack of knowledge nor the actual state of the technology.

Although commercial equipment can be successfully used to regulate velocity and sometimes regulate position, they cannot be satisfactorily used in robotics. In this situation, the controller must be specifically designed for the model that characterizes the dynamical behaviour of the system. The situation could be still more complex, when the parameters of the model are difficult to be quantified, of openly unknown. The study of this case was the motive for the development of the present analysis.

This approach is interesting from the application point of view, because we can add the compensation to an existing proportional-derivative controller.

The chapter is organized as follows. In Section 5.2, we present the dynamical model of a DC motor controlled by armature. In Section 5.3, we state the control problem. In Section 5.4, we mention some previous reported solutions. We present the control and adaptation algorithms, and the stability analysis in Section 5.5. In Section 5.6, we present some simulation results. We give some conclusions in Section 5.7, and the references are listed at the end of the document.

5.2 DYNAMICAL MODEL.

A classic description of a direct current motor controlled by armature, is given by the following equations [5]

$$
\begin{aligned}
J\ddot{q}(t) + f\dot{q}(t) &= \mathrm{K}_i I(t) \\
\mathrm{L}\dot{I}(t) + \mathrm{R}I(t) + e(t) &= v(t) \\
\mathrm{K}_b\dot{q}(t) &= e(t)
\end{aligned}
$$

where $q(t)$ is the angular position of axis, $v(t)$ is the armature voltage, $I(t)$ is the armature current, $e(t)$ is the induced voltage, J is the inertia of the rotor, f is the friction coefficient, K_i, K_b, R, and L are electric parameters of the motor.

Neglecting the armature inductance L, considering the armature $v(t)$ as input, and the angular position $q(t)$ as output, we get

$$q(t) = \frac{\mathrm{K}}{p(\tau p + 1)}v(t) \tag{5.1}$$

where p is the differential operator, thus $p{=}d/dt$, and

$$\mathrm{K} = \frac{\mathrm{K}_i}{\mathrm{R}f + \mathrm{K}_b\mathrm{K}_i} > 0$$

$$\tau = \frac{JR}{Rf + K_b K_i} > 0$$

Notice that K and τ depend on the inertia J and the friction coefficient f. It happens in applications of direct current motors, that the inertia is partially unknown, because it depends on the load connected to the motor. Also the friction coefficient varies according to the maintenance and lubrication of the motor. In this work we suppose that K and τ are unknown constants.

5.3 PROBLEM FORMULATION.

Consider the model of a direct current motor, where the parameters K and τ are unknown constants. Given a function $q_d(t)$, with derivatives of first and second order, called the desired angular position, we have to design a controller that provides $v(t)$ in such a way that the position error will converge to zero, that is

$$\lim_{t \to \infty} \tilde{q}(t) = 0$$

where $\tilde{q}(t)$ is the vector of position errors defined as

$$\tilde{q}(t) = q_d(t) - q(t)$$

It suffices to show that $\tilde{q}(t) \to 0$, to guarantee that $\dot{q}(t) \to \dot{q}_d(t)$, and $\ddot{q}(t) \to \ddot{q}_d(t)$. So this is our objective.

We do not expect that the controller will perform derivatives of the input signal, then we consider that the values of $q_d(t)$, $\dot{q}_d(t)$, and $\ddot{q}_d(t)$, can be provided to the controller. The most simple way of doing this is by the use of a reference model, where we specify a stable second order transfer function G(p), with relative degree of 2. The input is a reference function $q_r(t)$, and the output is $q_d(t)$, that is

$$q_d(t) = G(p)\ q_r(t) = \frac{b_o}{p^2 + a_1 p + a_o} q_r(t)$$

or equivalently

$$\frac{d}{dt}\begin{bmatrix} q_d(t) \\ \dot{q}_{d(t)} \end{bmatrix} = \begin{bmatrix} \dot{q}_d(t) \\ b_o q_r(t) - a_o q_d(t) - a_1 \dot{q}_d(t) \end{bmatrix} \tag{5.2}$$

In Figure 1 a block diagram is shown.

We also consider that the position $q(t)$ and the velocity $\dot{q}(t)$ are measurable. The control problem stated, can be solved by designing an adaptive controller with trajectory following objective, for a linear second order plant with relative degree of 2, and unknown constant parameters. We now present some solutions that have been previously reported.

5.4 LITERATURE REVIEW.

State feedback controller with adaptive compensation [2].

The structure of the controller is the following

$$\begin{aligned} v(t) &= K_p\tilde{q}(t) - K_v\dot{q}(t) + \theta_1\dot{q}_d(t) + \theta_2\ddot{q}_d(t) \\ \dot{\theta}_1 &= \gamma\dot{q}_d(t)\tilde{q}(t) \\ \dot{\theta}_2 &= \gamma\ddot{q}_d(t)\tilde{q}(t) \end{aligned}$$

where $\gamma > 0$ is the adaptation gain, K_p and K_v are design positive constants, called the position and velocity gains. The analysis presented shows that if γ is small enough, and $q_d(t)$ does not have high frequency components, the control objective is satisfied. However, this controller present parametric overflow in regulation.

PD controller with adaptive compensation [8,9].

This strategy is based on a robot controller. In the application to direct current motors, it results to be

$$\begin{aligned} v(t) &= K_p\tilde{q}(t) + K_v\dot{\tilde{q}}(t) + \theta_1(\ddot{q}_d(t) + \lambda\dot{\tilde{q}}(t)) + \theta_2(\dot{q}_d(t) + \lambda\tilde{q}(t)) \\ \dot{\theta}_1 &= \gamma(\ddot{q}_d(t) + \lambda\dot{\tilde{q}}(t))(\dot{\tilde{q}}(t) + \lambda\tilde{q}(t)) \\ \dot{\theta}_2 &= \gamma(\dot{q}_d(t) + \lambda\tilde{q}(t))(\dot{\tilde{q}}(t) + \lambda\tilde{q}(t)) \end{aligned}$$

where K_p, K_v, γ, and λ are design positive constants. It was shown that the closed–loop adaptive system is stable and satisfies the control objective. However, the structure is quite complex, and a large number of on-line computations are required.

Adaptive computed torque controller [3].

This controller is fundamented on the controller proposed in [3], for motion control of robot manipulators. When we apply it to DC motors, it is reduced to

$$
\begin{aligned}
v(t) &= \theta_1\left[K_p\tilde{q}(t) + K_v\dot{\tilde{q}}(t) + \ddot{q}_d(t)\right] + \theta_2\dot{q}(t) \\
\dot{\theta}_1 &= \gamma\left(\ddot{q}_d(t) + K_v\dot{\tilde{q}}(t) + K_p\dot{q}(t)\right)\nu \\
\dot{\theta}_2 &= \gamma\dot{q}(t)\nu \\
\nu &= \frac{p}{p+\lambda}\dot{\tilde{q}}(t) + \frac{1}{p+\lambda}\left(K_v\dot{\tilde{q}}(t) + K_p\tilde{q}(t)\right)
\end{aligned}
$$

where K_p, K_v, γ, and λ are design positive constants. As in the other controllers, it was shown that the closed–loop adaptive system is stable and the control objective is satisfied. Notice that the structure is still quite complex.

5.5 PD PLUS ADAPTIVE FEEDFORWARD COMPENSATION.

Of above controllers, the most simple is the state feedback controller with adaptive compensation. A disadvantage of this controller is that, it is conditioned to an adequate selection of γ, and the type of reference functions $q_d(t)$ is also restricted. In the last two controllers this does not happen, but they are much more complex. In this section we propose a simple controller that satisfies the control objective. Although, we do not show convergence of the parametric error to zero, we found in simulations, that it does.

We consider that a minimum value of K is known, say K_{min}, and also a maximum value of τ is known, say $\tau_{\max}$. In practice this is not restrictive.

The structure of the proposed controller is

$$v(t) = K_p\tilde{q}(t) + K_v\dot{\tilde{q}}(t) + \theta_1\dot{q}_d(t) + \theta_2\ddot{q}_d(t) \tag{5.3}$$

$$\dot{\theta}_1 = \gamma \dot{q}_d(t)\left(\dot{\tilde{q}}(t) + \lambda \tilde{q}(t)\right) \tag{5.4}$$
$$\dot{\theta}_2 = \gamma \ddot{q}_d(t)\left(\dot{\tilde{q}}(t) + \lambda \tilde{q}(t)\right) \tag{5.5}$$

where K_p, K_v, and γ, are design positive constants. We select λ in such a way that the following inequality is satisfied

$$0 < \lambda < \frac{1 + K_{\min} K_v}{\tau_{\max}} \tag{5.6}$$

The selected value of λ satisfies

$$0 < \lambda < \frac{1 + K K_v}{\tau} \tag{5.7}$$

which implies that

$$0 < (1 + K K_v - \tau\lambda)\lambda$$

also

$$0 < (1 + K K_v)\lambda - \tau\lambda^2 + K K_v \tag{5.8}$$

since K and K_v are positive.

The block diagram that represents the closed–loop system for the proposed controller is shown in Figure 2. Notice that if $q_d(t)$ is constant, this controller is reduced to the proportional plus velocity feedback controller given by

$$v(t) = K_p \tilde{q}(t) - K_v \dot{q}(t)$$

for which it is well known [10] that satisfies the control objective, and it is exponentially stable. Also it is robust against additive perturbations in the input or in the output.

We define the parametric error $\tilde{\theta}(t) \in \mathbf{R}^2$ as

$$\tilde{\theta}(t) = \begin{bmatrix} \tilde{\theta}_1(t) \\ \tilde{\theta}_2(t) \end{bmatrix} = \begin{bmatrix} \theta_1 - 1/K \\ \theta_2 - \tau/K \end{bmatrix} \tag{5.9}$$

Notice that because K and τ are constants, $\dot{\tilde{\theta}}_1(t) = \dot{\theta}_1$ and $\dot{\tilde{\theta}}_2(t) = \dot{\theta}_2$. The closed–loop state equation is given by

$$\frac{d}{dt}\begin{bmatrix}\tilde{q}(t)\\ \dot{\tilde{q}}(t)\\ \tilde{\theta}_1(t)\\ \tilde{\theta}_2(t)\end{bmatrix} = \begin{bmatrix}\dot{\tilde{q}}\\ -\frac{1}{\tau}\left((1+\mathrm{K}K_v)\dot{\tilde{q}}+\mathrm{K}K_p\tilde{q}+\mathrm{K}(\tilde{\theta}_1\dot{q}_d+\tilde{\theta}_2\ddot{q}_d)\right)\\ \gamma\dot{q}_d(\dot{\tilde{q}}+\lambda\tilde{q})\\ \gamma\ddot{q}_d(\dot{\tilde{q}}+\lambda\tilde{q})\end{bmatrix} \tag{5.10}$$

which is a nonautonomous differential equation, that has one equilibrium in the origin.

Consider the following Lyapunov candidate function

$$\mathrm{V}(\tilde{q},\dot{\tilde{q}},\tilde{\theta}) = \frac{1}{2}\left[\tau\left(\dot{\tilde{q}}+\lambda\tilde{q}\right)^2 + \left(\mathrm{K}K_p+\lambda(1+\mathrm{K}K_v)-\tau\lambda^2\right)\tilde{q}^2 + \frac{\mathrm{K}}{\tau}\tilde{\theta}^T\tilde{\theta}\right] \tag{5.11}$$

It is easy to check that $\mathrm{V}(\tilde{q},\dot{\tilde{q}},\tilde{\theta})$ is a positive definite function, and it is also decrescent [10]. Its derivative with respect to time is given by

$$\dot{\mathrm{V}}(\tilde{q},\dot{\tilde{q}},\tilde{\theta}) = -(1+\mathrm{K}K_v-\tau\lambda)\dot{\tilde{q}}^2 - \lambda\mathrm{K}K_p\tilde{q}_2 \leq 0 \tag{5.12}$$

which guarantees that the equilibrium is stable, and $\mathrm{V}(\tilde{q},\dot{\tilde{q}},\tilde{\theta})$ is a Lyapunov function. Thus, since it is decrescent, positive definite, and also radially unbounded, its arguments $\tilde{q}$, $\dot{\tilde{q}}$, and $\tilde{\theta}$ are bounded, that is

$$\tilde{q},\dot{\tilde{q}} \in L_\infty, \qquad \tilde{\theta} \in L_\infty^2 \tag{5.13}$$

On the other hand, we can obtain the following relation from (13)

$$\mathrm{V}\left(\tilde{q}(0),\dot{\tilde{q}}(0),\tilde{\theta}(0)\right) \geq (1+\mathrm{K}K_v-\tau\lambda)\int_o^\infty \dot{\tilde{q}}(s)^2 ds + \lambda\mathrm{K}K_p\int_o^\infty \tilde{q}(s)^2 ds \tag{5.14}$$

which implies that $\tilde{q},\dot{\tilde{q}} \in L_2$. We now present the following lemma.

Lemma 5.1 [9] (pp. 232)

Let $f:\mathbf{R}_+ \to \mathbf{R}$. If $f \in L_2$ and $\dot{f} \in L_\infty$, then $f(t) \to 0$ as $t \to \infty$.

This lemma guarantees that

$$\lim_{t\to\infty} \tilde{q}(t) = 0$$

It has been shown that the control objective is satisfied.

5.6 SIMULATION RESULTS.

We now present simulation results of the proposed adaptive scheme . The numeric values of the parameters of the motor were τ=0.1 and K=1. The initial conditions were $\tilde{q}(0) = 2$, $\dot{\tilde{q}}(0) = 0$, $\tilde{\theta}_1(0) = 2$, and $\tilde{\theta}_2(0) = 2$. The parameters of the controller were K_p=1/4, K_v=1, $\gamma = 1/2$, and $\lambda = 10$. Notice that λ satisfies (7)

$$0 < \lambda < \frac{1 + KK_v}{\tau} = 20$$

The desired trajectory is $q_d(t)$=sin(t).

In Figure 3, both $q(t)$ and $q_d(t)$ are shown. Notice that by t=100, $q(t)$ has converged to $q_d(t)$.

In Figure 4, it can be seen that the position error vanishes.

We present the state trajectory in figure 5. Notice that it converges to zero.

Although in this analysis we did not prove that the parametric error vanishes, we have evidence provided by the simulation, to conclude that it does. Notice that both $\tilde{\theta}_1(t)$ and $\tilde{\theta}_2(t)$ converge to zero.

With this simulations, we show that this controller satisfies the control objective, thus it can be successfully applied to control DC motors that have (5.1) as dynamical model.

We have built an analog electronic card to implement this controller, and we found that it is valuable.to have a compensation that can be added to an existing PD controller for DC motors. These experimental results are not presented in this document.

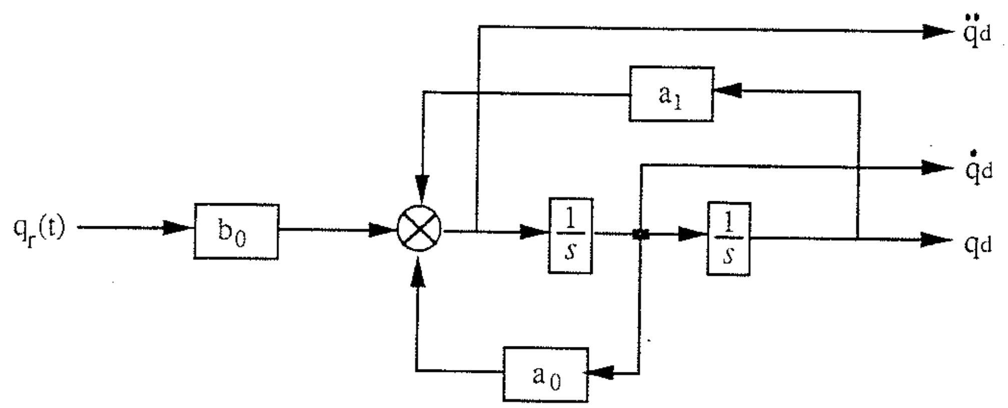


Figure 1. Reference Model.

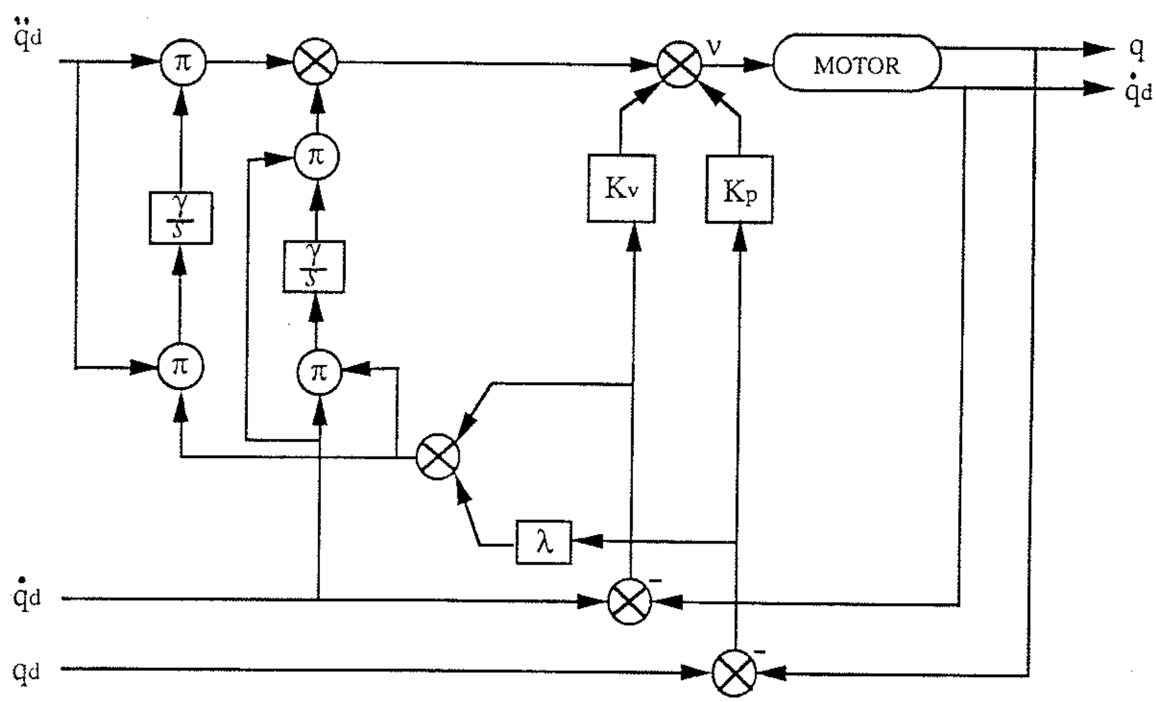


Figure 2. Block Diagram.

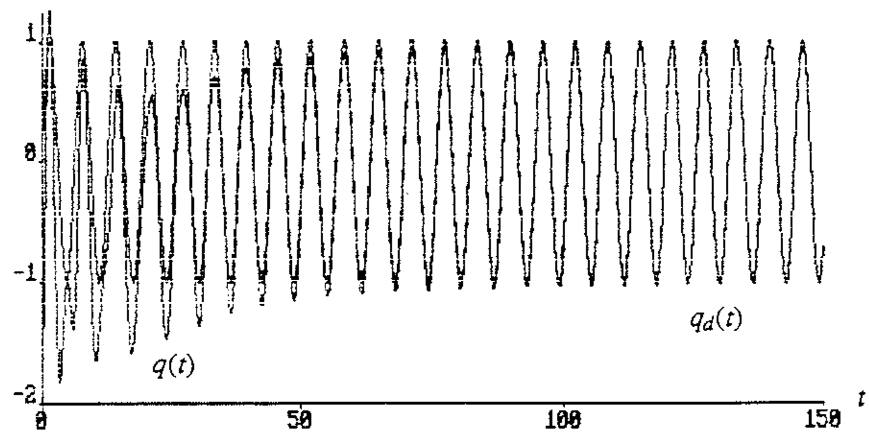


Figure 3. Position and Reference.

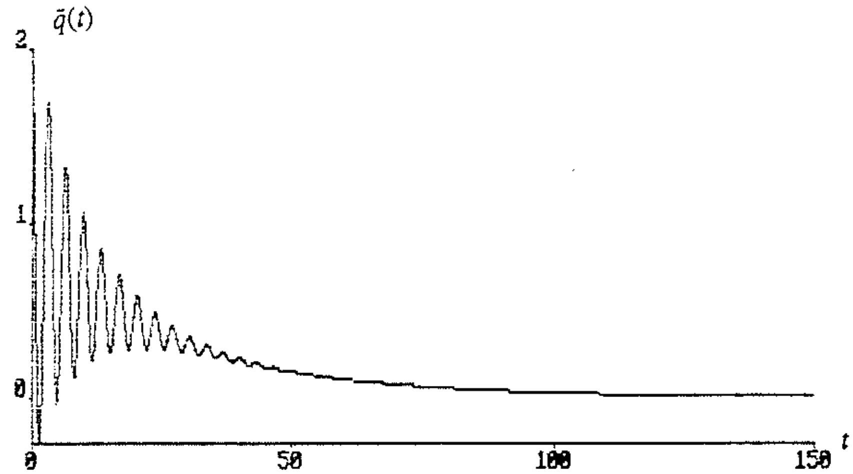


Figure 4. Position Error.

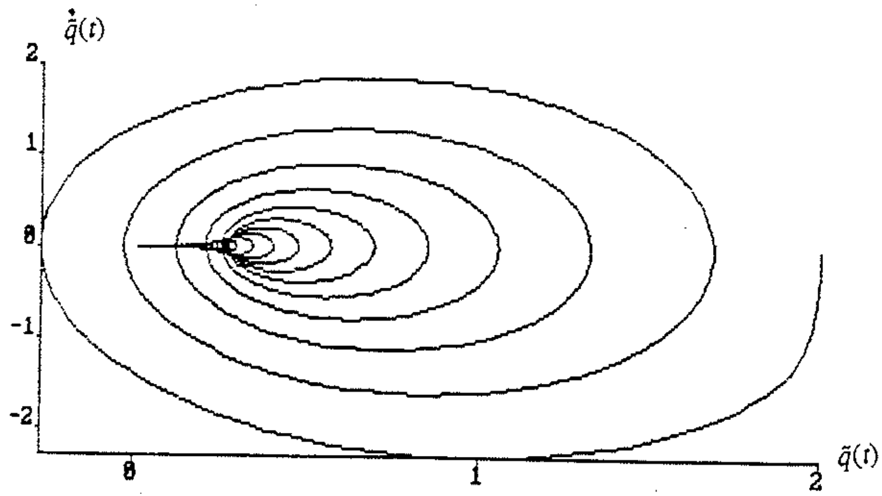
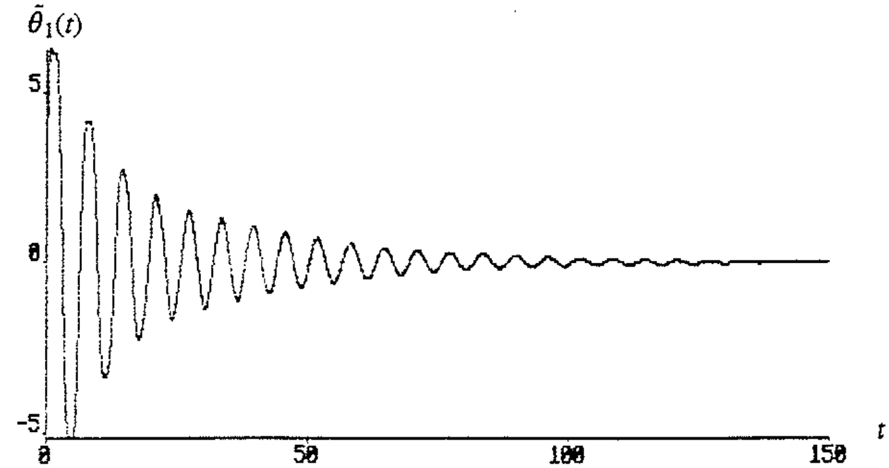


Figure 5. State Trajectory.



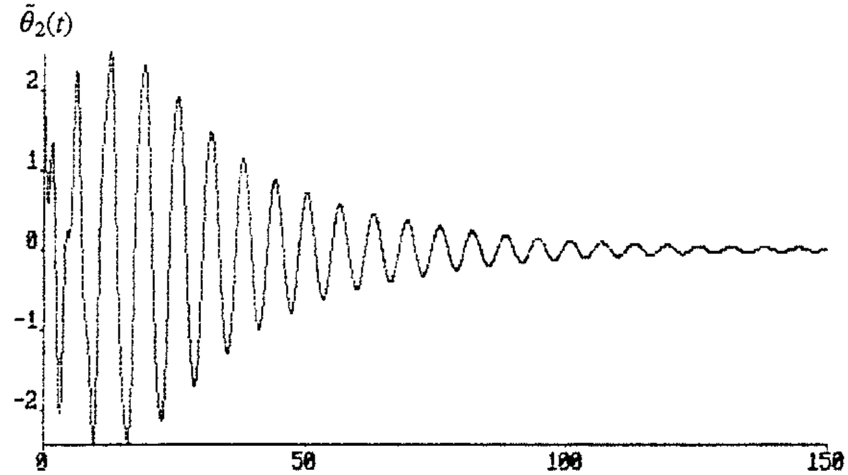


Figure 6. Parametric Error.

5.7 CONCLUSIONS.

In this work we studied the control problem for DC motors with parametric uncertainties, for position and velocity specifications. We have proposed a PD with an adaptive feedforward compensation, which guarantees stability and achievement of the control objective.

The design of the controller requires the values of some parametric bounds of the motor, which are easy to obtain in practice, from the technical specification of the motor

Simulation results were presented, showing that the algorithm performs well.

We have proposed a relatively simple compensation for PD controllers for DC motors, that can be easily added to an existing controller.

REFERENCES.

[1] Desoer, C., Vidyasagar, M., *Feedback Systems: Input–Output Properties*, Academic Press, New York, 1975.

[2] Kelly, R., "A Linear–State Feedback plus Adaptive Feedforward Control for DC Servo–Motors", IEEE Trans. on Industrial Electronics, vol. IE–34, no. 2, May, 1987.

[3] Kelly, R., "Adaptive Computed Torque plus Compensation Control for Robot Manipulators", Mechanism and Machine Theory, vol. 25, no 2, 1990.

[4] Kelly, R., Rueda, A., Alcorta, E., "PD Controller plus Adaptive Feedforward Compensation for Direct Current Motors", Proceedings of the 3rd International Conference on Advances on Comunications and Control, Victoria, Canada, October, 1991, (pp. to be anounced).

[5] Kuo, B., *Automatic Control Systems*, 4th edition, Prentice Hall, New York, 1982.

[6] Narendra, K., Annaswamy, A., *Stable Adaptive Systems*, Prentice Hall, New York, 1989.

[7] Sastry, S., Bodson. M., *Adaptive Control, Stability, Convergence and Robustness*, Prentice Hall, New Jersey, 1989.

[8] Slotine, J., Li., W., "On the Adaptive Control of Robot Manipulators", The International Journal of Robotics Research, vol. 6, No. 3, Fall, 1987.

[9] Slotine, J., Li., W., "Adaptive Manipulator Control: a Case of Study", IEEE Transactions on Automatic Control, vol. 33, no. 11, November 1988.

FINAL CONCLUSIONS
AND FUTURE RESEARCH

In this work, the basic issues concerning the modelling and control of robot manipulators have been presented. Our efforts resulted in putting together several important results in a clear and compact manner. First of all, we introduce some mathematical tools that are important in our analysis. Then, a kinematic analysis has been described, in where we developed expressions for the Jacobian. A detailed dynamic analysis has been presented, ending up with the dynamical model of a robot manipulator of n degrees of freedom. Some important properties useful in control were also studied. Future research will be done in the area of motion planning and flexible links.

It has been been shown that the PD plus computed feedforward controller is stable under certain conditions, that we found by studying the closed–loop equation and the behaviour of its equilibrium points. We showed that the trajectory following objective is satisfied. We confirmed our theoretical results with simulations using a manipulator of one degree of freedom. We will try to solve the problem of force control objective in future work.

A new PD–type adaptive controller has been proposed for independent–joint control of robot manipulators. We showed that this controller has a more simple structure than previous reported controllers of the same type. We solved the problem of parametric uncertainties in the model of the actuator, when this actuator is a direct current motor or a direct drive motor. From our simulations we found that in every case the parameters of the motor were identified. Future work will be done trying to solve the problem of time–varying parameters, using adaptive control.

New techniques like fuzzy controllers and neural networks identifiers will be also studied in future work, as an attemp to provided alternative solutions to those previously reported using conventional techniques.