POLITECNICO DI MILANO

Corso di Laurea Magistrale in Ingegneria Meccanica

Dipartimento di Meccanica

Model Predictive Parametric Control of a Non-Holonomic Mobile Manipulator: a kinodynamic hybrid approach

Relatore: Professor Luca Bascetta

Tesi di Laurea di:

FRANCESCO GENNARI, matricola 884546

RAFFAELE COLOMBO, matricola 874779

Anno Accademico 2018-2019

Dedica lavoro ....

Contents

Aknowledgements

Grazie a tutti

Abstract

Industry is increasingly going towards the complete automatization of processes, using, among others, mobile manipulators for transporting parts between machines and work stations in a time-optimal manner and therefore increasing the efficiency of the plant. USC (University of Southern California) has proposed in S.Thakar et al.,2018 an algorithm for the definition of optimal trajectories for mobile manipulator pick-and-transport operations: the robot arm picks up the part during the motion of the mobile base, avoiding the latter to stop to allow the operation. Once done the planning of the motion of the mobile manipulator, USC asked us to face the problem of its control.

In our work we aimed to a control which could drive properly all the degrees of freedom of the robot during its motion coping also with unexpected obstacles and events and unprecise positioning during grasping operation. The control logic we chose for this task is Model Predictive Control. The difficulty of its implementation lies in the high non-linearity of the system, modelling it both kinematically or dynamically. In order to decrease the computational effort of the optimizer it has been used a linear time-varying model-based predictive controller (LTV-MPC) parametrizing the input profile computed by the optimizer. In this way the controller doesn't have to find the optimal inputs for all the timesteps in the control horizon, but only some parameters to let the input follow a certain curve. Moreover to further simplify the definition of the problem avoiding the definition of the terminal constraints and terminal cost of the MPC problem, which where necessary for the stability of the control, we used a monotonically increasing weighting profile of the running costs in the cost function, following the idea of M.Alamir,2017.

The controller has been tested on the robot sending command messages to

ROS through Simulink nodes. This way the controller could have been written in Matlab, using the Matlab-Interface of IPOPT as optimizer. It has been seen that this controller . . . .
It is useful for . . . . . .
Next steps should be . . . .

Chapter 1

Introduction

Mobile Manipulators

In the last decades industry is being moving from the use of traditional robotic arms (manipulators), which executes highly specific simple tasks in controlled and secure environments, like spot welding, towards more complex and flexible robotic systems.
Nowadays different types of robots can be found in a great variety of environments, ranging from the cleaning robots in people houses, to the exploration robots sent in space.
In industry, some of the reasons why automated systems are used are:

- A better production quality can be reached having the possibility to more easily measure and control the performances;
- Automation can also improve the productivity, reducing downtimes and the consumption of resources;
- Robots can have access to environments which are too risky or not accessible for humans;
- Robots maintenance can be much cheaper than other man-driven alternatives.

The mobility of robots substantially increased what they can cope. Robots are now expected to explore unknown dynamic environments, interact with

human beings or manipulate hazardous products. Mobile manipulators are systems that combine locomotion and manipulation capabilities in order to fulfil such missions.
Mobile manipulators can be very different, according to the environment where they are supposed to work.

- Submarine mobile manipulators for underwater applications;

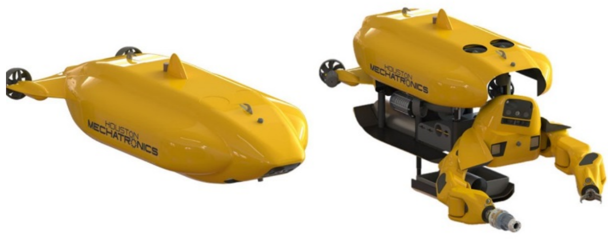


Figure 1.1: Houston Mechatronics's Aquanaut

- legged humanoid or animal-like robots for service missions;

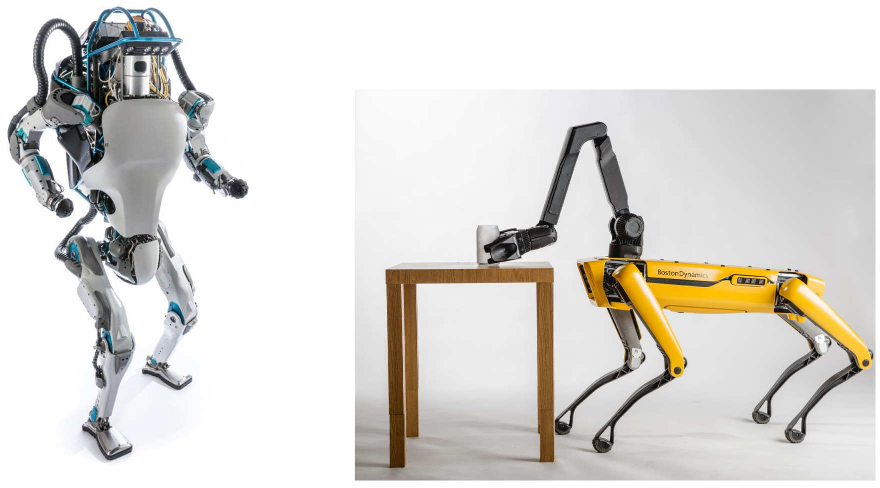


Figure 1.2: Boston Dynamics's Atlas and Spot Mini

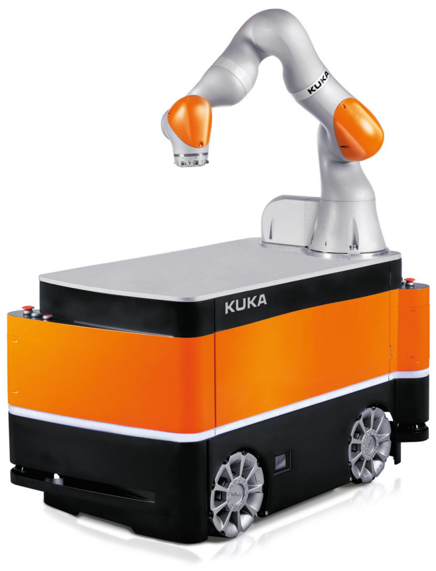


Figure 1.3: Kuka KMR iiwa

- The most popular ones are the wheeled mobile manipulators or WMM, which combine a wheeled mobile base and one or more robotic arms.

Among the wheeled mobile manipulators there are many different types, but usually they have the following characteristics:

1. The mobile base is a nonholonomic system;

2. The entire system is often kinematically redundant with respect to the task to be achieved;

3. The dynamic properties of the base are very different from the manipultor ones.

Since it is important for the understanding of the problem, the concept of Nonholonomy is now explained.

Holonomic and Nonholonomic constraints

Considering a system of N rigid bodies and assuming that all of them can reach any position in the space, in order to find uniquely the position of all the points of the system, it is necessary to define a vector of $6N = m$ parameters. This set of parameters is termed *generalized coordinates of the unconstrained system.* Constraints acting on the position (configuration) of the system are said to be *holonomic.* Holonomic constraints are expressed in the form:

$$h\left(x, q\right) = 0 \tag{1.1}$$

where h is a vector of dimension $s < m$. The constraints that constrain the velocities of the system of rigid bodies but not their positions are called *nonholonomic.* In general all constraints which are nonintegrable are said to be nonholonomic, so also inequality constraints about the configuration are considered nonholonomic. Nonholonomic constraints reduce the mobility of the mechanical system in a completely different way with respect to holonomic constraints. The fact that the constraint is nonintegrable means that there is no loss of accessibility to the all configuration space for the system. In other words, while the number of DOFs decreases due to the constraint, the number of generalized coordinates cannot be reduced, not even locally. Only the subspace of the generalized velocities is reduced. Nonholonomic constraints are expressed in the form:

$$h(x, \dot{x}, q) = 0 \tag{1.2}$$

where h is a vector of dimension $s < m$. On the assumption that h has continuous and continuously differentiable components, and its Jacobian $\frac{\partial h}{\partial x}$ has full rank, the constraints equations allow the elimination of s out of m coordinates of the system. With the remaining $n = m - s$ coordinates it is possible to determine uniquely the configurations of the system satisfying the constraints. Such coordinates are the *generalized coordinates of the constrained system.*

Kinematic constraints both holonomic or nonholonomic are usually written in the Pfaffian form, i.e. linearly with respect to the generalized velocities.

$$a_i^T(q)\,\dot{q} = 0 \qquad i = 1, ..., k < n \tag{1.3}$$

$$A^T(q)\,\dot{q} = 0$$

Talking about mobile robots, wheels are by far the most common mechanism to achieve locomotion. Any wheeled vehicle is subject to kinematic constraints that reduce in general its local mobility, while leaving intact the possibility of reaching arbitrary configurations by appropriate manoeuvres. The pure rolling constraint for a wheel is expressed in the Pfaffian form as

$$\dot{x}\sin\theta - \dot{y}\cos\theta = \begin{bmatrix} \sin\theta & cos\theta & 0 \end{bmatrix} \dot{q} = 0 \tag{1.4}$$

and entails that, in the absence of slipping, the velocity of the contact point has zero component in the direction orthogonal to the sagittal plane. The angular velocity of the disk around the vertical axis instead is unconstrained. This constraint is nonholonomic, because it implies no loss of accessibility in the configuration space of the disk.

Chapter 2

Model of the Mobile Manipulator

Mobile Robot Model

Kinematics

Assuming to have a n dimensional, k constraint equations imply that the admissible generalized velocities of the system for any of its configurations q necessarily belong to a $(n-k)$-dimensional space. Specifically, expressing the constraint equations in Pfaffian form, the generalized velocities should belong to the null space of matrix A^T.
Thus, it can be written:

$$\dot{q} = \sum_{j=1}^{m} g_j(q)u_j = G(q)u \qquad m = n - k \tag{2.1}$$

Where $q \in \mathbb{R}^n$ is the state vector, $u = \begin{bmatrix} u_1 & \cdots & u_m \end{bmatrix}^T \in \mathbb{R}^m$ is the input vector and $\{g_1(q) \cdots g_m(q)\}$ is a basis of the null space of A^T, named *input vector fields*.
The choice of the input vector fields is not unique, since the input u can have different meanings: velocities, torques or others.

Unicycle Model

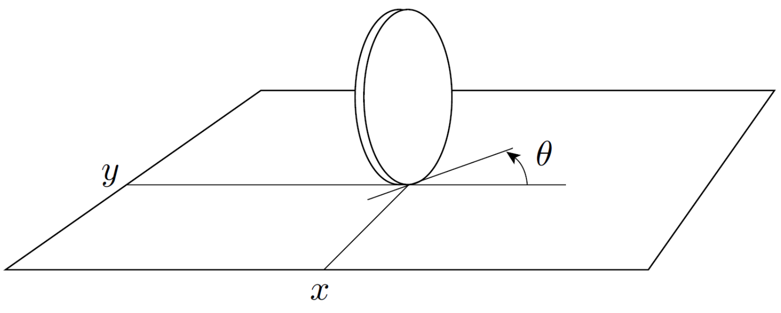


Figure 2.1: Unicycle model

A unicycle is a vehicle which configuration is completely described by the state vector $q = [xy\theta]^T$, where x and y are the Cartesian coordinates of the contact point of the wheel with the ground or of the wheel centre and θ is the orientation of the wheel with respect to the x axis.
As we already seen in the previous chapter, the constraint of pure rolling for the unicycle is expressed as:

$$\dot{x}\sin\theta - \dot{y}\cos\theta = \begin{bmatrix}\sin\theta & cos\theta & 0\end{bmatrix}\dot{q} = 0 \tag{2.2}$$

$$A^T(q)\,\dot{q} = 0$$

This equation expresses that the velocity of the contact point (or of the wheel centre) is zero in the direction orthogonal to the sagittal axis of the disk.
Hence in the unicycle case $n = 3$ and $k = 1$, so the dimension of the basis of the null space of A^T is 2. Choosing:

$$G(q) = \begin{bmatrix}g_1(q) & g_2(q)\end{bmatrix} = \begin{bmatrix}\cos\theta & 0\\ \sin\theta & 0\\ 0 & 1\end{bmatrix} \tag{2.3}$$

All the generalized velocities of the unicycle at q are a combination of $g_1(q)$

and $g_2(q)$.

$$\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega \tag{2.4}$$

Where v is the driving velocity, i.e. the velocity of the disk along its sagittal plane, and ω is the steering velocity, i.e. the angular speed around its vertical axis.
A lot of mobile robots are kinematically equivalent to a unicycle, like differential drive robots, skid steering robots or synchro drive vehicles.
In most of the unicycle-like mobile robots, v and ω are the command input to the system, while the actual velocity input are ω_R and ω_L, i.e. the right and left wheels angular speeds.

$$v = \frac{r\left(\omega_R + \omega_L\right)}{2} \qquad \omega = \frac{r\left(\omega_R - \omega_L\right)}{d} \tag{2.5}$$

$$\omega_R = \frac{1}{r}\left(v + \frac{d}{2}\omega\right) \qquad \omega_L = \frac{1}{r}\left(v - \frac{d}{2}\omega\right) \tag{2.6}$$

Where r is the wheels radius and d is the distance between their centres.

Bicycle Model

A bicycle is a vehicle having an orientable wheel and a fixed wheel. Kinematically it can be described with the generalized coordinates $q = [xy\theta\phi]^T$, where x and y are the Cartesian coordinates of the centre of the rear wheel, θ is the orientation of the vehicle with respect to the x axis, and ϕ is the steering angle of the front wheel with respect to the vehicle axis.
Both wheels are subject to pure rolling constraints. So expressing the zero velocity of the centre of the front wheel in the direction orthogonal to the wheel itself, and the zero velocity of the centre of the rear wheel in the direction orthogonal to the sagittal axis of the vehicle:

$$\dot{x}_f \sin\left(\theta + \phi\right) - \dot{y}_f \cos\left(\theta + \phi\right) = 0 \tag{2.7}$$

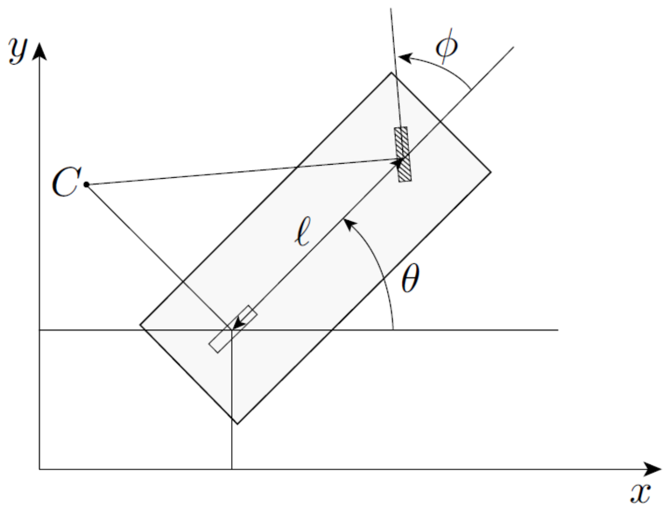


Figure 2.2: Bicycle model

$$\dot{x}\sin\theta - \dot{y}\cos\theta = 0 \tag{2.8}$$

Where:

$$x_f = x + \ell\cos\theta$$
$$y_f = y + \ell\sin\theta$$

Are the cartesian coordinates of the centre of the front wheel.
The lines orthogonal to the two wheels meet in the instantaneous centre of rotation C.
The pure rolling constraints written in the above equation can be expressend in the Pfaffian form as:

$$A^T(q)\,\dot{q} = 0$$

$$A^T(q) = \begin{bmatrix} \sin\theta & -\cos\theta & 0 & 0 \\ \sin(\theta+\phi) & -\cos(\theta+\phi) & -\ell\cos\phi & 0 \end{bmatrix} \tag{2.9}$$

Following the same reasoning done before, we have 4 degrees of freedom and 2 kinematic constraints, so the dimension of the null space of $A(q)^T$ is

$n - k = 2$. An example of basis of the null space is:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\phi \\ \sin\theta\cos\phi \\ \sin\phi/\ell \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \omega \tag{2.10}$$

If the vehicle is front-wheel drive, or:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \tan\phi/\ell \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \omega \tag{2.11}$$

If the vehicle is rear-wheel drive.

Dynamics

The dynamic model of a mobile robot can be derived using the Lagrangian formulation for a n-dimensional system, defining the Lagrangian L of the mechanical system as the difference between kinetic and potential energy.

$$\mathcal{L}(q, \dot{q}) = \mathcal{T}(q, \dot{q}) - \mathcal{U}(q) = \frac{1}{2}\dot{q}^T B(q)\dot{q} - \mathcal{U}(q) \tag{2.12}$$

Where $B(q)$ is the inertia matrix, which is symmetric and positive definite. Writing the Lagrange equations:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}}\right) - \left(\frac{\partial \mathcal{L}}{\partial q}\right)^T = S(q)\tau + A(q)\lambda \tag{2.13}$$

In this formulation we are adding the energetical term due to the kinematic constraints adding the *Lagrange multipliers* term. Substituting the Lagrangian term, we obtain the dynamic model of the mobile base:

$$B(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) = S(q)\tau + A(q)\lambda \tag{2.14}$$

$$A^T(q)\,\dot{q} = 0$$

Where:

$$C(q,\dot{q}) = \dot{B}(q)\dot{q} - \frac{1}{2}\left(\frac{\partial}{\partial q}\left(\dot{q}^T B(q)\dot{q}\right)\right)^T \tag{2.15}$$

$$g(q) = \frac{\partial \mathcal{U}(q)}{\partial q} = 0 \tag{2.16}$$

The potential energy of a mobile base during its motion can be considered constant, so it is possible to disregard it.
It is possible to further simplify the dynamic model exploiting a useful property of the kinematic relation 2.1. Since we defined the coloums of $G(q)$ as vectors belonging to the null space of $A(q)^T$, the following relation holds:

$$A(q)^T G(q) = 0 \qquad \text{or} \qquad G(q)^T A(q) = 0 \tag{2.17}$$

So, premultiplying the dynamic model by $G(q)^T$:

$$G^T(q)\left(B(q)\ddot{q} + C(q,\dot{q})\dot{q}\right) = G^T(q)S(q)\tau \tag{2.18}$$

Reducing the n differential equations to a system of m differential equations without Lagrange multipliers.

Unicycle Model

In a unicycle case, like a differential drive robot (Figure 2.3), the kinetic energy can be expressed as:

$$\mathcal{T}(q,\dot{q}) = \frac{1}{2}M\dot{x}_{COG}^2 + \frac{1}{2}M\dot{y}_{COG}^2 + \frac{1}{2}J\dot{\theta}^2 \tag{2.19}$$

Where:

$$x_{COG} = x + x_C\cos\theta - y_C\sin\theta$$
$$y_{COG} = y + x_C\sin\theta + y_C\cos\theta$$

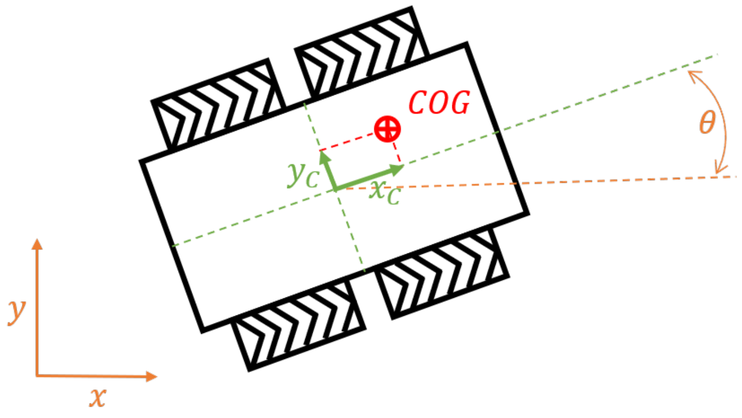


Figure 2.3: Scheme of a Differential Drive Mobile Robot

Are the global coordinates of the centre of gravity of the mobile base, and their velocities are:

$$\dot{x}_{COG} = \dot{x} - \dot{\theta} x_C \sin\theta - \dot{\theta} y_C \cos\theta$$
$$\dot{y}_{COG} = \dot{y} + \dot{\theta} x_C \cos\theta - \dot{\theta} y_C \sin\theta$$

And M and J are respectively the base's mass and moment of inertia along its vertical axis z.

So, after substituting these equations in the Lagrange equation we can write:

$$B(q) = \begin{bmatrix} M & 0 & -M\left(x_C \sin\theta + y_C \cos\theta\right) \\ 0 & M & M\left(x_C \cos\theta - y_C \sin\theta\right) \\ -M\left(x_C \sin\theta + y_C \cos\theta\right) & M\left(x_C \cos\theta - y_C \sin\theta\right) & J + M x_C^2 + M y_C^2 \end{bmatrix}$$

$$C(q, \dot{q}) = \begin{bmatrix} 0 & 0 & M\dot{\theta}\left(y_C \sin\theta - x_C \cos\theta\right) \\ 0 & 0 & -M\dot{\theta}\left(y_C \cos\theta + x_C \sin\theta\right) \\ 0 & 0 & 0 \end{bmatrix} \tag{2.20}$$

Taking the kinematic equation of the unicycle model, deriving it and substituting it in the reduced dynamic model we obtain:

$$\dot{q} = G(q) \begin{bmatrix} v_{long} \\ \omega \end{bmatrix} = G(q) v \tag{2.21}$$

$$\ddot{q} = \dot{G}(q)v + G(q)\dot{v} \tag{2.22}$$

$$\tilde{B}(q)\dot{v} + \tilde{C}(q,\dot{q})v = \tilde{S}(q)\tau \tag{2.23}$$

Where:

$$\begin{aligned}
&\tilde{B}(q) = G^T(q)B(q)G(q)\\
&\tilde{C}(q,\dot{q}) = G^T(q)C(q,\dot{q})G(q) + G^T(q)B(q)\dot{G(q)}\\
&\tilde{S}(q) = G^T(q)S(q)
\end{aligned}$$

In this way it is also possible to express the system in *state space form*:

$$\begin{bmatrix}\dot{q}\\ \dot{v}\end{bmatrix} = \begin{bmatrix}[0] & G(q)\\ [0] & -\tilde{B}^{-1}\tilde{C}\end{bmatrix}\begin{bmatrix}q\\ v\end{bmatrix} + \begin{bmatrix}[0]\\ \tilde{B}^{-1}\tilde{S}\end{bmatrix}\tau \tag{2.24}$$

All the system can be eventually simplified considering the centre of gravity in the geometric centre of the mobile robot.

$$x_C = 0 \quad , \quad y_C = 0;$$

$$B(q) = \begin{bmatrix}M & 0 & 0\\ 0 & M & 0\\ 0 & 0 & J\end{bmatrix} \qquad C(q,\dot{q}) = \begin{bmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix} \tag{2.25}$$

$$\tilde{B}(q)\dot{v} = \tilde{S}(q)\tau \tag{2.26}$$

So, the state space dynamic model becomes:

$$\begin{bmatrix}\dot{q}\\ \dot{v}\end{bmatrix} = \begin{bmatrix}[0] & G(q)\\ [0] & [0]\end{bmatrix}\begin{bmatrix}q\\ v\end{bmatrix} + \begin{bmatrix}[0]\\ \tilde{B}^{-1}\tilde{S}\end{bmatrix}\tau \tag{2.27}$$

Manipulator Model

Kinematics

A manipulator consists of a series of rigid links connected by joints. In order to plan the manipulator motion from an initial to a final configuration, i.e. the set of joint values, it is necessary to define its pose with respect to a reference frame.

Rototraslation Matrices

An orthonormal reference frame **x',y',z'** can be described with respect to an orthonormal reference frame **x,y,z** in the following way:

$$\mathbf{x'} = x'_x\mathbf{x} + x'_y\mathbf{y} + x'_z\mathbf{z} \tag{2.28}$$

$$\mathbf{y'} = y'_x\mathbf{x} + y'_y\mathbf{y} + y'_z\mathbf{z} \tag{2.29}$$

$$\mathbf{z'} = z'_x\mathbf{x} + z'_y\mathbf{y} + z'_z\mathbf{z} \tag{2.30}$$

These three equations can be expressed as a rotation matrix of the frame

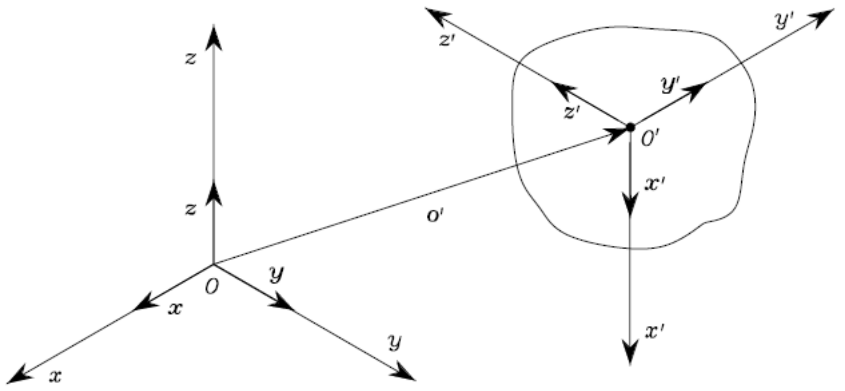


Figure 2.4

x',y',z' with respect to the frame **x,y,z**:

$$R = \begin{bmatrix} \mathbf{x'} & \mathbf{y'} & \mathbf{z'} \end{bmatrix} = \begin{bmatrix} x'_x & y'_x & z'_x \\ x'_y & y'_y & z'_y \\ x'_z & y'_z & z'_z \end{bmatrix} \tag{2.31}$$

Consider now a point P whose coordinates are referred to frame **x,y,z** describe the vector **p**, and the coordinates referred to the rotated frame **x',y',z'** describe the vector **p'**. Then it is possible to write:

$$\mathbf{p} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \qquad \mathbf{p'} = \begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix} \tag{2.32}$$

$$\mathbf{p} = p'_x\mathbf{x'} + p'_y\mathbf{y'} + p'_z\mathbf{z'} = \begin{bmatrix} \mathbf{x'} & \mathbf{y'} & \mathbf{z'} \end{bmatrix} \mathbf{p'} = R\mathbf{p'} \tag{2.33}$$

The rotation matrix is orthogonal because its columns are mutually orthog-

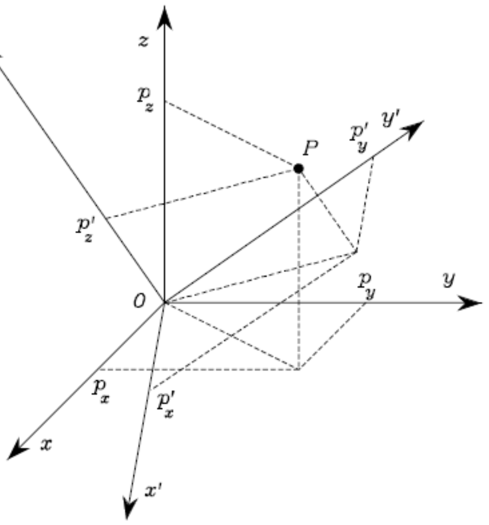


Figure 2.5

onal, by definition.

$$R^T R = I \qquad R^T = R^{-1} \tag{2.34}$$

If we then consider two different reference frames: x^0, y^0, z^0 and x^1, y^1, z^1 with two different origins O^0 and O^1, a point P is represented by the two frames respectively with vectors p^0 and p^1. The vector p^0 can be represented

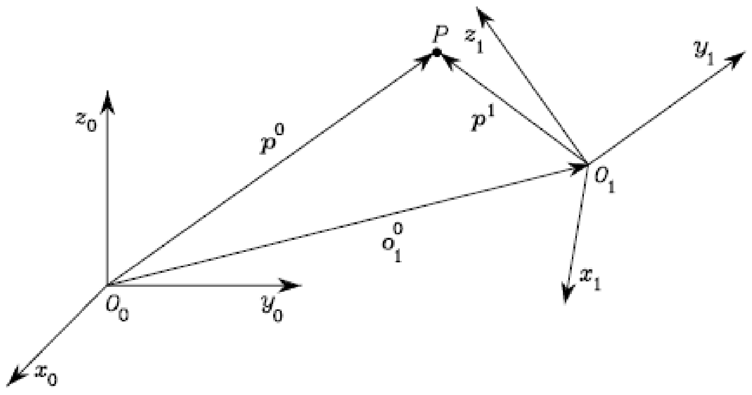


Figure 2.6

as the sum of two vectors:

$$\mathbf{p^0} = \mathbf{o_1^0} + R_1^0\mathbf{p^1} \tag{2.35}$$

Where R_1^0 is the rotation matrix of frame x^1, y^1, z^1 with respect to frame x^0, y^0, z^0 and so $R_1^0\mathbf{p^1}$ is the representation of vector $\mathbf{p^1}$ in the frame x^0, y^0, z^0. It is possible to express the equation above in a compact form using the homogenous transformation matrix simply defining:

$$\tilde{\mathbf{p}} = \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} \qquad A_1^0 = \begin{bmatrix} R_1^0 & \mathbf{o_1^0} \\ [0] & 1 \end{bmatrix} \tag{2.36}$$

$$\tilde{\mathbf{p}}^0 = A_1^0\tilde{\mathbf{p}}^1 \tag{2.37}$$

The matrix A_1^0 is called *rototraslation matrix.*
Inverting the equation 2.35 we can express the inverse transformation.

$$\mathbf{p^1} = -R_0^1\mathbf{o_1^0} + R_0^1\mathbf{p^0} \tag{2.38}$$

$$A_0^1 = \begin{bmatrix} R_0^1 & -R_0^1 \mathbf{o_1^0} \\ [0] & 1 \end{bmatrix} \tag{2.39}$$

Where:

$$R_0^1 = (R_1^0)^{-1} = (R_1^0)^T$$

$$A_0^1 = (R_1^0)^{-1}$$

Both with rotation and rototraslation matrices, it is possible to obtain a composition of consecutive transformations by postmultiplying the matrices related to the single rotations:

$$R_2^0 = R_1^0 R_2^1 \tag{2.40}$$

$$A_2^0 = A_1^0 A_2^1 \tag{2.41}$$

Denavit-Hartenberg Parameters

Since the first ‘80s a popular approach to describe the kinematics of robots, and therefore the rototraslation matrices to express the transformation from one joint frame to the following, is the convention introduced by Jacques Denavit and Richard Hartenberg in 1955 for a compact description of mechanisms. Their representation is based on the concept of the common normal between joint axes, defined as the shortest line between two axes, perpendicular to both axes.
The convention firstly assigns to each link a reference frame referring to the following rules:

- Axis z_i is chosen along the axis of joint i+1;
- The origin O_i is located at the intersection of axis z_i with the common normal to axes z_{i-1} and z_i;
- Axis x_i is chosen as an extension of the common normal to axes z_{i-1} and z_i;
- Axis y_i is chosen to complete a right-handed frame.

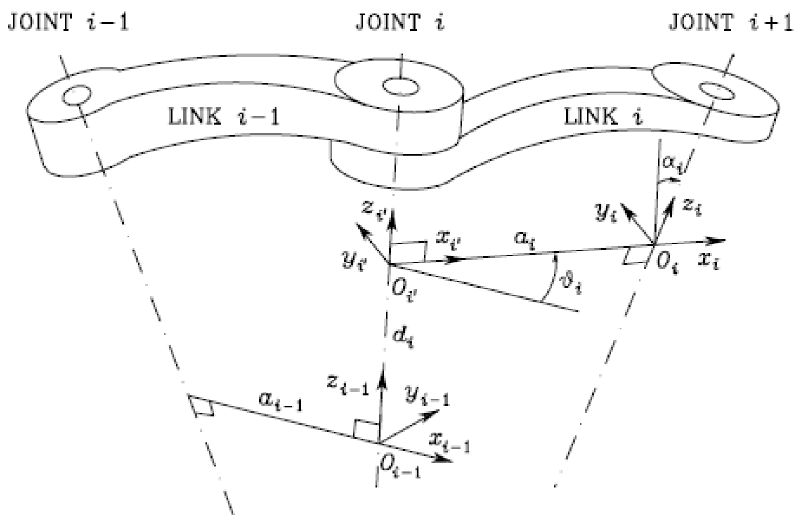


Figure 2.7: Denavit-Hartenberg convention for Robot links

Once established a frame for each link, 4 parameters describe the pose of frame i with respect to frame i-1.

- a_i : distance between z_i and z_{i-1}. Only geometrical, depends only on the geometry of the links
- d_i : coordinate of O_i along z_{i-1}. Varies if the joint is prismatic.
- α_i: angle between axes z_{i-1} and z_i (positive counterclockwise with respect to axis x_i). Only geometrical, depends only on the geometry of the links.
- θ_i: angle between axes x_{i-1} and x_i (positive counterclockwise with respect to axis z_{i-1}). Varies if the joint is revolute.

The transformation matrix between frame i and frame i-1 can then be built:

$$A_i^{i-1}(q_i) = \begin{bmatrix} \cos\theta & -\sin\theta\cos\alpha & \sin\theta\sin\alpha & a\cos\theta \\ \sin\theta & \cos\theta\cos\alpha & -\cos\theta\sin\alpha & a\sin\theta \\ 0 & \sin\alpha & \cos\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.42}$$

This matrix is function only on the joint variables θ and d.

Direct and Inverse Kinematics

As anticipated, a manipulator is basically made of links chained together by joints that connect the *base* of the manipulator to its *end effector*.
The *direct kinematics* of the manipulator express the position and orientation of the end effector as a function of the robot configuration. Using the concepts showed up to this point, this corresponds to the definition of the rototraslation matrix that describes the end effector frame with respect to the global reference frame:

$$A_{ee}^{base} = A_1^{base} A_2^1 A_3^2 \cdots A_{ee}^{ee-1} \tag{2.43}$$

The opposite problem is the definition of the *inverse kinematics*, i.e. given the position and orientation of the end effector, finding the corresponding joint variables. The inverse kinematics problem is not easy to solve since it is often nonlinear and there can be infinite solutions as well as none.
An important concept which roboticists are still working on is *kinematic redundancy*, i.e. when the number of joint variables is greater than the dimension of the operational space, which is 6 in case of the definition of the 3D position and orientation of the end effector. When a system is kinematically redundant, direct kinematics is always possible to be defined, while inverse kinematics can have infinite solutions. Solving the *kinematic redundancy problem* means to define a criterion in order to be able to pick one among the infinite configurations which have the end effector in the same position and orientation.
This concept will be very important in the defintion of Mobile Manipulator control.

Dynamics

As we already seen in the case of mobile robots the dynamics of a robot can be expressed with the Euler-Lagrange formulation.

$$\mathcal{L} = \mathcal{T} - \mathcal{U}$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}}\right) - \left(\frac{\partial \mathcal{L}}{\partial q}\right)^T = \tau$$

Where q is the vector of the generalized coordinates and τ is the vector of the generalized forces.
The potential energy of the manipulator is given by

$$\mathcal{U} = \sum_{i=1}^{n} (\mathcal{U}_{l_i} + \mathcal{U}_{m_i}) \tag{2.44}$$

Where $\mathcal{U}_{l_i}$ and $\mathcal{U}_{m_i}$ are the potential energy due to the gravitational forces of the links and of the motors respectively.
Similarly, the kinetic energy of the entire system is the sum of the kinetic energies of the links and of the motors

$$\mathcal{T} = \sum_{i=1}^{n} (\mathcal{T}_{l_i} + \mathcal{T}_{m_i}) \tag{2.45}$$

Beginning with the expression of the links kinetic energy:

$$\mathcal{T}_{l_i} = \frac{1}{2}\int_{V_{l_i}} \dot{\mathbf{p}}{*}_i^T \dot{\mathbf{p}}{*}_i \rho dV \tag{2.46}$$

Where $\dot{\mathbf{p}}{*}_i$ the velocity vector of the elementary particle of link i.
In order to relate the velocities of every elementary particle of the manipulator to the joint velocities we have to express in a proper way the Jacobians.

Differential Kinematics

First of all we need to introduce the skew symmetric matrix $S(\boldsymbol{\omega})$. It can be obtained from the differentiation with respect to time of a rotation matrix:

$$R(t)R(t)^T = I \tag{2.47}$$

$$\dot{R}(t)R(t)^T + R(t)\dot{R}(t)^T = 0 \tag{2.48}$$

S is defined as:

$$S(t) = \dot{R}(t)R(t)^T \tag{2.49}$$

It is possible now to define the derivative of the rotation matrix as:

$$\dot{R}\,(t) = S\,(t)\,R(t) \tag{2.50}$$

$$p\,(t) = R\,(t)\,p' \tag{2.51}$$

$$\dot{p}\,(t) = \dot{R}\,(t)\,p' = S(t)R\,(t)\,p' \tag{2.52}$$

But from mechanics it is also known that:

$$\dot{p}\,(t) = \boldsymbol{\omega}(t) \times R\,(t)\,p' \tag{2.53}$$

Where $\boldsymbol{\omega}\,(t) = \begin{bmatrix}\omega_x & \omega_y & \omega_z\end{bmatrix}$ is the vector of the angular velocity of the rotating frame described by $R(t)$. So S can be interpreted as an operator depending only on $\boldsymbol{\omega}\,(t)$.

$$S\,(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & -\omega_x & 0 \end{bmatrix} \tag{2.54}$$

Jacobians

In order to express the Lagrange equation as a function of the joint variables velocities we now introduce the Jacobian matrices for each link.

$$\dot{\mathbf{p}}_{l_i} = j_{P_1}^{(l_i)}\dot{q}_1 + j_{P_2}^{(l_i)}\dot{q}_2 + \cdots + j_{P_i}^{(l_i)}\dot{q}_i = \mathcal{J}_P^{(l_i)}\dot{q} \tag{2.55}$$

$$\omega_{l_i} = j_{O_1}^{(l_i)}\dot{q}_1 + j_{O_2}^{(l_i)}\dot{q}_2 \cdots + j_{O_i}^{(l_i)}\dot{q}_i = \mathcal{J}_O^{(l_i)}\dot{q} \tag{2.56}$$

Where with $\mathcal{J}_P^{(l_i)}$ we mean the Jacobian matrix relating the joint velocities to the linear velocity of the centre of mass of link i, and with $\mathcal{J}_O^{(l_i)}$ we mean the Jacobian matrix relating the joint velocities to the angular velocity of the centre of mass of link i. The columns of the Jacobian matrices are:

$$j_{P_j}^{(l_i)} = \begin{cases} z_{j-1} & \text{for a } \textit{prismatic} \text{ joint} \\ z_{j-1} \times (\mathbf{p}_{l_i} - \mathbf{p}_{j-1}) & \text{for a } \textit{revolute} \text{ joint} \end{cases} \tag{2.57}$$

$$j_{O_j}^{(l_i)} = \begin{cases} 0 & \text{for a } \textit{prismatic} \text{ joint} \\ z_{j-1} & \text{for a } \textit{revolute} \text{ joint} \end{cases} \tag{2.58}$$

Inertia Matrix

Recovering the kinetic energy expression, we can now relate the velocities $\dot{\mathbf{p}*}_i$ to the velocity vector $\dot{\mathbf{p}}_{l_i}$ of the centre of mass of the link.

$$\dot{\mathbf{p}*}_i = \dot{\mathbf{p}}_{l_i} + \boldsymbol{\omega}_i \times \mathbf{r}_i = \dot{\mathbf{p}}_{l_i} + S(\boldsymbol{\omega}_i)\mathbf{r}_i \tag{2.59}$$

Substituting this expression in the equation of the kinetic energy and exploiting the properties of the integral of the centre of mass we obtain.

$$\mathcal{T}_{l_i} = \frac{1}{2}m_{l_i}\dot{\mathbf{p}}_{l_i}^T\dot{\mathbf{p}}_{l_i} + \frac{1}{2}\boldsymbol{\omega}_i{}^T R_i I_{l_i}^i R_i^T \boldsymbol{\omega}_i \tag{2.60}$$

$$\mathcal{T}_{l_i} = \frac{1}{2}m_{l_i}\dot{q}^T \mathcal{J}_P^{(l_i)^T} \mathcal{J}_P^{(l_i)}\dot{q} + \frac{1}{2}\dot{q}^T \mathcal{J}_O^{(l_i)^T} R_i I_{l_i}^i R_i^T \mathcal{J}_O^{(l_i)}\dot{q} \tag{2.61}$$

In an analogous way it can be computed also the motors kinetic energy.

$$\mathcal{T}_{m_i} = \frac{1}{2} m_{m_i} \dot{\mathbf{p}}_{m_i}^T \dot{\mathbf{p}}_{m_i} + \frac{1}{2} \boldsymbol{\omega}_{m_i}{}^T I_{m_i} \boldsymbol{\omega}_{m_i} \tag{2.62}$$

Where $\dot{\mathbf{p}}_{m_i}$ is the velocity vector of the centre of mass of the motor and $\boldsymbol{\omega}_{m_i}$ is the angular velocity of the rotor of motor i.

$$\mathcal{T}_{m_i} = \frac{1}{2} m_{m_i} \dot{q}^T \mathcal{J}_P^{(m_i)^T} \mathcal{J}_P^{(m_i)} \dot{q} + \frac{1}{2} \dot{q}^T \mathcal{J}_O^{(m_i)^T} R_{m_i} I_{m_i}^{m_i} R_{m_i}^T \mathcal{J}_O^{(m_i)} \dot{q} \tag{2.63}$$

Summing all the contributions of the single links, the total kinetic energy of the manipulator is:

$$\mathcal{T} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij}(q) \dot{q}_i \dot{q}_j = \frac{1}{2} \dot{q}^T B(q) \dot{q} \tag{2.64}$$

Where:

$$\begin{aligned} B(q) = \sum_{i=1}^{n} & m_{l_i} \mathcal{J}_P^{(l_i)^T} \mathcal{J}_P^{(l_i)} + \frac{1}{2} \mathcal{J}_O^{(l_i)^T} R_i I_{l_i}^i R_i^T \mathcal{J}_O^{(l_i)} \\ & + m_{m_i} \mathcal{J}_P^{(m_i)^T} \mathcal{J}_P^{(m_i)} + \mathcal{J}_O^{(m_i)^T} R_{m_i} I_{m_i}^{m_i} R_{m_i}^T \mathcal{J}_O^{(m_i)} \end{aligned}$$

Is the $(n \times n)$ symmetric positive-definite inertia matrix of the system.

Coriolis Matrix

Recovering the Lagrange equation:

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}}\right)^T - \left(\frac{\partial \mathcal{L}}{\partial q}\right)^T = \tau$$

$$\begin{aligned} \frac{d}{dt}\left(\frac{\partial \mathcal{T}}{\partial \dot{q}_i}\right) - \left(\frac{\partial \mathcal{T}}{\partial q_i}\right) &= \sum_{j=1}^{n} b_{ij}(q) \ddot{q}_j + \sum_{j=1}^{n} \sum_{k=1}^{n} \frac{\partial b_{ij}(q)}{\partial q_k} \dot{q}_k \dot{q}_j \\ & \quad - \frac{1}{2} \sum_{j=1}^{n} \sum_{k=1}^{n} \frac{\partial b_{jk}(q)}{\partial q_i} \dot{q}_k \dot{q}_j \\ &= \sum_{j=1}^{n} b_{ij}(q) \ddot{q}_j + \sum_{j=1}^{n} \sum_{k=1}^{n} h_{ijk}(q) \dot{q}_k \dot{q}_j \end{aligned} \tag{2.65}$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{T}}{\partial \dot{q}}\right)^T - \left(\frac{\partial \mathcal{T}}{\partial q}\right)^T = B(q)\ddot{q} + C\left(q,\dot{q}\right)\dot{q} \tag{2.66}$$

Where $C\left(q,\dot{q}\right)$ is the non-symmetric positive definite Coriolis matrix that represents the effect of motion of joint j and k on the motion of joint i.

Gravitational Vector

Continuing considering the potential energy within the Lagrange equation.

$$\mathcal{U} = \sum_{i=1}^{n} \left(\mathcal{U}_{l_i} + \mathcal{U}_{m_i}\right)$$

$$\mathcal{U}_{l_i} = -m_{l_i}\mathbf{g}_0^T\mathbf{p}_{l_i} \quad , \quad \mathcal{U}_{m_i} = -m_{m_i}\mathbf{g}_0^T\mathbf{p}_{m_i} \tag{2.67}$$

Where $\mathbf{g}_0$ is the gravity acceleration vector. So:

$$\frac{\partial \mathcal{U}}{\partial q_i} = -\sum_{j=1}^{n}\left(m_{l_i}\mathbf{g}_0^T\frac{\partial \mathbf{p}_{l_i}}{\partial q_i} + m_{m_i}\mathbf{g}_0^T\frac{\partial \mathbf{p}_{m_i}}{\partial q_i}\right) = g_i(q) \tag{2.68}$$

So the final equation of the manipulator dynamic model is:

$$B\left(q\right)\ddot{q} + C\left(q,\dot{q}\right)\dot{q} + g\left(q\right) = \ \tau \tag{2.69}$$

Where $g(q)$ is the gravitational vector that represents the moments acting on links due to gravity force.

Mobile Manipulator Model

Considering the general case of a mobile manipulator composed by a manipulator arm mounted on a mobile platform, its kinematic model set the location of the end effector as a function of both the arm and base general-

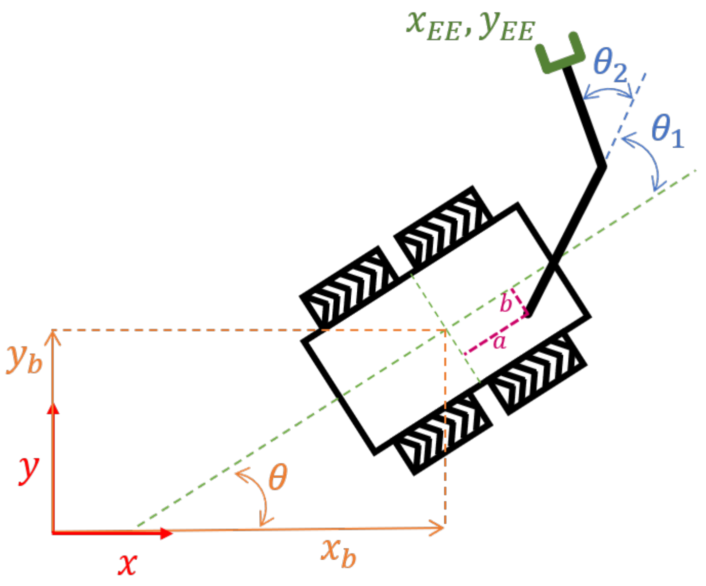


Figure 2.8: A simple Mobile Manipulator composed by a mobile base and a 2-DoF robotic arm

ized coordinates.

$$X_{EE} = \begin{bmatrix} p_{EE} \\ \Phi \end{bmatrix} = f(q) = f\left(\begin{bmatrix} q_a \\ q_b \end{bmatrix}\right) \tag{2.70}$$

Where $p_{EE} = \begin{bmatrix} x_{EE} & y_{EE} & z_{EE} \end{bmatrix}^T$ is the vector of the end effector position and$\Phi = \begin{bmatrix} \vartheta_{EE} & \varphi_{EE} & \psi_{EE} \end{bmatrix}$ is the vector of the end effector orientation that can be expressed with different conventions like Azimut-Elevation-Rotation or Roll-Pitch-Yaw. It is possible to define the function f of Eq.2.70 adding the rototranslation matrix A^{global}_{base} relative to the mobile base motion, to the direct kinematics of the arm 2.43.

$$A^{global}_{base} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & x_b + \sqrt{a^2+b^2}\cos\theta \\ \sin\theta & \cos\theta & 0 & y_B + \sqrt{a^2+b^2}\sin\theta \\ 0 & 0 & 1 & h_{base} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.71}$$

So the forward kinematics can be obtained consequently:

$$A^{global}_{ee} = A^{global}_{base} A^{base}_1 A^1_2 \cdots A^{ee-1}_{ee} \tag{2.72}$$

Differentiating f we can obtain the differential forward kinematics of the Mobile Manipulator.

$$\dot{X}_{EE} = \frac{\partial f}{\partial q_a}(q_a, \theta)\dot{q}_a + \frac{\partial f}{\partial q_b}(q_a, \theta)\dot{q}_b \tag{2.73}$$

Where $\frac{\partial f}{\partial q_a}$ is the same as described in section 2.2.2, and $\frac{\partial f}{\partial q_b}(q_a, \theta)$ can be obtained at the same way considering x and y as prismatic joints and θ as a revolute joint of the compound system. In doing this the nonholonomic constraint for the mobile platform should be taken into account: As we had said before the forward kinematics of a system is always solvable, but for redundant systems like mobile manipulators the inverse kinematics problem has infinite solutions.

$$B\left(q\right)\ddot{q} + C\left(q, \dot{q}\right)\dot{q} + g\left(q\right) = \tau \tag{2.74}$$

Mixed Kinematics and Dynamics Model

The idea of the following model is to describe kinematically the mobile base and dynamically the manipulator, in order to have as inputs:

- the torques applied at the manipulator joints
- the longitudinal and angular velocity of the mobile base.

For this purpose, the kinematic equation of the mobile base will be used within the dynamic model of the whole mobile manipulator. We will notate the variables in the following way.

Mobile base coordinates: $x = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$

Manipulator coordinates: $q = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_6 \end{bmatrix}$

Vector of longitudinal and angular velocity of the base: $v = \begin{bmatrix} v_{long} \\ \omega \end{bmatrix}$

Writing down the dynamic equation of the entire mobile manipulator, which can be obtained as described in section 2.2.2:

$$B(x,q)\begin{bmatrix}\ddot{x}\\ \ddot{q}\end{bmatrix} + C(x,\dot{x},q,\dot{q})\begin{bmatrix}\dot{x}\\ \dot{q}\end{bmatrix} + g(x,q) = \begin{bmatrix}\tau_x\\ \tau_q\end{bmatrix} + \begin{bmatrix}A(x)\\ 0\end{bmatrix}\lambda \tag{2.75}$$

$$\begin{aligned} &\begin{bmatrix}B_x(x,q) & B_{xq}(x,q)\\ B_{qx}(x,q) & B_q(x,q)\end{bmatrix}\begin{bmatrix}\ddot{x}\\ \ddot{q}\end{bmatrix} \\ &+\begin{bmatrix}C_x(x,\dot{x},q,\dot{q}) & C_{xq}(x,\dot{x},q,\dot{q})\\ C_{qx}(x,\dot{x},q,\dot{q}) & C_q(x,\dot{x},q,\dot{q})\end{bmatrix}\begin{bmatrix}\dot{x}\\ \dot{q}\end{bmatrix} \\ &+\begin{bmatrix}0\\ g(q)\end{bmatrix} = \begin{bmatrix}\tau_x\\ \tau_q\end{bmatrix} + \begin{bmatrix}A(x)\\ 0\end{bmatrix}\lambda \end{aligned} \tag{2.76}$$

Premultiplying by G^T only the rows relative to the base coordinates:

$$\begin{aligned} &\begin{bmatrix}G^T(x)B_x(x,q) & G^T(x)B_{xq}(x,q)\\ B_{qx}(x,q) & B_q(x,q)\end{bmatrix}\begin{bmatrix}\ddot{x}\\ \ddot{q}\end{bmatrix} \\ &+\begin{bmatrix}G^T(x)C_x(x,\dot{x},q,\dot{q}) & G^T(x)C_{xq}(x,\dot{x},q,\dot{q})\\ C_{qx}(x,\dot{x},q,\dot{q}) & C_q(x,\dot{x},q,\dot{q})\end{bmatrix}\begin{bmatrix}\dot{x}\\ \dot{q}\end{bmatrix} \\ &+\begin{bmatrix}0\\ g(q)\end{bmatrix} = \begin{bmatrix}G^T(x)\tau_x\\ \tau_q\end{bmatrix} \end{aligned} \tag{2.77}$$

The columns of G are a basis of the null space of $A(x)T$, so the Lagrange multipliers term is set to zero. If we now consider the kinematic equation of the base 2.1:

$$\begin{bmatrix}\dot{x}\\ \dot{q}\end{bmatrix} = \begin{bmatrix}G(x) & 0\\ 0 & I\end{bmatrix}\begin{bmatrix}v\\ \dot{q}\end{bmatrix} \tag{2.78}$$

Whose derivative is:

$$\ddot{x} = G(x)\dot{v} + \dot{G}(\dot{x},x)v \tag{2.79}$$

$$\begin{bmatrix}\ddot{x}\\ \ddot{q}\end{bmatrix} = \begin{bmatrix}G(x) & 0\\ 0 & I\end{bmatrix}\begin{bmatrix}\dot{v}\\ \ddot{q}\end{bmatrix} + \begin{bmatrix}\dot{G}(\dot{x},x) & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}v\\ \dot{q}\end{bmatrix} \tag{2.80}$$

If we substitute these expressions in the dynamic model of the mobile manipulator:

$$\begin{bmatrix} G^T B_x G & G^T B_{xq} \\ B_{qx} G & B_q \end{bmatrix} \begin{bmatrix} \dot{v} \\ \ddot{q} \end{bmatrix} + \begin{bmatrix} G^T B_x \dot{G} + G^T C_x G & G^T C_{xq} \\ B_{qx} \dot{G} + C_{qx} G & C_q \end{bmatrix} \begin{bmatrix} v \\ \dot{q} \end{bmatrix} + \begin{bmatrix} 0 \\ g \end{bmatrix} = \begin{bmatrix} G^T \tau_x \\ \tau_q \end{bmatrix} \tag{2.81}$$

Where the parenthesis have not been transcribed in order to have a compact expression. What now we are interested in is the dynamic equation of only the manipulator part, i.e. the second row of 2.81.

$$\begin{bmatrix} B_{qx} G & B_q \end{bmatrix} \begin{bmatrix} \dot{v} \\ \ddot{q} \end{bmatrix} + \begin{bmatrix} B_{qx} \dot{G} + C_{qx} G & C_q \end{bmatrix} \begin{bmatrix} v \\ \dot{q} \end{bmatrix} + g = \tau_q \tag{2.82}$$

v, which is the velocity vector of the base, is the input vector of the kinematic model of the base. The idea is to use v and $\dot{v}$ as inputs also in the dynamic model of the manipulator. Rearranging the previous equation we obtain:

$$[B_q]\,\ddot{q} + [C_q]\,\dot{q} + g = \tau_q - [B_{qx} G]\,\dot{v} - \left[B_{qx} \dot{G} + C_{qx} G\right] v \tag{2.83}$$

Which in state space form is:

$$\begin{aligned} \begin{bmatrix} \ddot{q} \\ \dot{q} \end{bmatrix} &= \begin{bmatrix} -B_q{}^{-1} C_q & 0 \\ I & 0 \end{bmatrix} \begin{bmatrix} \dot{q} \\ q \end{bmatrix} + \begin{bmatrix} -B_q{}^{-1} g \\ 0 \end{bmatrix} + \begin{bmatrix} B_q{}^{-1} \\ 0 \end{bmatrix} \tau_q \\ &+ \begin{bmatrix} -B_q{}^{-1} B_{qx} G \\ 0 \end{bmatrix} \dot{v} \\ &+ \begin{bmatrix} -B_q{}^{-1} \left[B_{qx} \dot{G} + C_{qx} G\right] \\ 0 \end{bmatrix} v \end{aligned} \tag{2.84}$$

Adding also the kinematic equation of the base we have the entire model of the system in state space form:

$$\begin{aligned}
\begin{bmatrix}\ddot{q}\\ \dot{q}\\ \dot{x}\end{bmatrix} &= \begin{bmatrix}-{B_q}^{-1}C_q & 0 & 0\\ I & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\dot{q}\\ q\\ x\end{bmatrix} + \begin{bmatrix}-{B_q}^{-1}g\\ 0\\ 0\end{bmatrix} + \begin{bmatrix}{B_q}^{-1}\\ 0\\ 0\end{bmatrix}\tau_q\\
&+ \begin{bmatrix}-{B_q}^{-1}B_{qx}G\\ 0\\ 0\end{bmatrix}\dot{v}\\
&+ \begin{bmatrix}-{B_q}^{-1}\left[B_{qx}\dot{G} + C_{qx}G\right]\\ 0\\ G\end{bmatrix}v
\end{aligned} \tag{2.85}$$

More precisely:

$$\begin{aligned}
\begin{bmatrix}\ddot{q}\\ \dot{q}\\ \dot{x}\end{bmatrix} &= \begin{bmatrix}-{B_q}^{-1}(x,q)C_q(x,\dot{x},q,\dot{q}) & 0 & 0\\ I & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\dot{q}\\ q\\ x\end{bmatrix} + \begin{bmatrix}-B_q(x,q)^{-1}g(q)\\ 0\\ 0\end{bmatrix}\\
&+ \begin{bmatrix}B_q(x,q)^{-1}\\ 0\\ 0\end{bmatrix}\tau_q\\
&+ \begin{bmatrix}-B_q(x,q)^{-1}B_{qx}(x,q)G(x)\\ 0\\ 0\end{bmatrix}\dot{v}\\
&+ \begin{bmatrix}-B_q(x,q)^{-1}\left[B_{qx}(x,q)\dot{G(x)} + C_{qx}(x,\dot{x},q,\dot{q})G(x)\right]\\ 0\\ G(x)\end{bmatrix}v
\end{aligned} \tag{2.86}$$

Chapter 3

Control Theory

The interconnection between control theory and robotics has a history that of over half a century during which control theory has developed solutions problems in robotics field and problems in robotics have gained the new control theories. In the last decade or two, the Progress in robotic systems has been rapid, especially thinking about the applications increase, from machine tool industry to biomedical applications for example. The early design of robotic systems was to develop mechanisms to be as stiff as possible modeled as single-input/single-output (SISO) linear systems with each joint controlled independently. Then Point-to-point control used to perform simple tasks such as spot welding or pick and place operations. Encoureged by industry request, more complex tasks such as arc welding and spray painting has been enabled from Continuous-path tracking. But to consider more advanced tasks like assembly was restricted by the limited or nonexistent sensing of the external environment. Higher speeds and higher payload-to-weight ratios required a better understanding of modeling of nonlinear dynamical systems. For this reason new theoretical results in nonlinear, robust, and adaptive control enabled more advanced applications. Today, robot control systems are highly advanced with integrated sensing systems. Robot networks, surgical robots, mobile robots, underwater and flying robots, and others are starting to play important roles in society.

PID Control

PID architecture is the most common form of feedback controllers. The first implementation of PID controllers dates back to early 20th century. Today more than 90 % of the control loops are PID type and are found in many areas. Nowdays is often combined with other advanced control techniques where generally the main usage of PID based controllers is for low level loops like motor controllers, integrated circuits etc. while advanced techniques are used for high level control. When dealing with mobile robots, which are generally designed to perform tasks more or less autonomously, the aim of the controller can be manifold. In the case of mobile robots for pick and place operation the architecture of the online controller is generally hierarchic and the role of the PID regulator is to track desired high level control input.

The general control law for a PID controller is defined as follows:

$$p(t) = K_P e(t) + K_I \int_0^t e(\tau) d\tau + K_D \frac{de(t)}{dt} \tag{3.1}$$

The advantages of this basic control logic is the simplicity of implementation, lucid meaning, and the near-zero memory usage from the phisical hardware. Because of that and the many tuning techniques developed in years PID as been widely accepted in industry. Anyway this basic feedback control law present consisten limitation for andvanced applications. The goal of this type of controllers is to define a control law able to reduce the errors computed by means of the given measured data, so it works going after the real system measured outputs. The tuning of the controller is made chosing the parameters K_P, K_I and K_D that change the response of the closed loop system. Those parameters have to be chosen properly in order to achive stability and performances. Nevertheless, it's generally not possible to take into account any changes in the system dynamic. Because of that other control techniques have been developed in years.

Inverse Dynamics Control

The inverse dynamics control approach is directly related to the solution of the inverse dynamics problem of the system. Given the specified motion and the desired properties of the resulting system, the control inputs that ensure stability and realization of these control objectives are to be found. By appropriately inverting the dynamic model of the system to be controlled, a control law can be found. Generally this control law is chosen in order to cancel out the nonlinear part of the dynamics, to decouple the interactions between the regulated variables, and specify the behaviour of the convergence of the error. The inverse dynamics controller is able to enforce the execution of prescribed motion of the system and at the same time to control the interaction forces with the environment. If we consider a generic dynamical system in the form:

$$H(q)\ddot{q} + C(q,\dot{q})\dot{q} + \tau_g(q) = \tau \tag{3.2}$$

The inverse dynamics control law can be defined:

$$\tau = H(q)v + C(q,\dot{q})\dot{q} + \tau_g(q) \tag{3.3}$$

From this control law results that $\ddot{q} = v$. Where v is a new conrol input, one approach to define it is with a PD (Proportional-Derivative) feedback:

$$v = \ddot{q}_d + K_V\dot{e}_q + K_P e_q \tag{3.4}$$

The overall control input becomes:

$$\tau = H(q)(\ddot{q}_d + K_V\dot{e}_q + K_P e_q) + C(q,\dot{q})\dot{q} + \tau_g(q) \tag{3.5}$$

So the resulting error dynamics is linear:

$$\ddot{e}_q + K_V\dot{e}_q + K_P e_q = 0 \tag{3.6}$$

That is, properly choosing K_V and K_P parameters theconvergence of the error is zero.

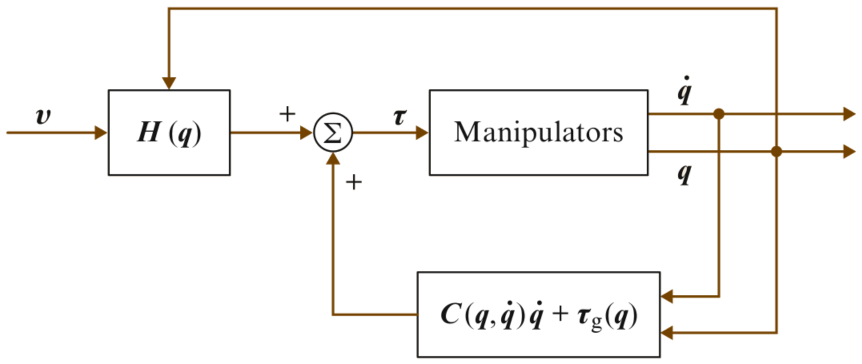

$\boldsymbol{v}$
$\boldsymbol{H}(\boldsymbol{q})$
+
Σ
+
$\boldsymbol{\tau}$
Manipulators
$\dot{\boldsymbol{q}}$
$\boldsymbol{q}$
$\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau}_{\mathrm{g}}(\boldsymbol{q})$

Figure 3.1: Inverse dynamics control scheme

The usage of the inverse dynamics in Robot controller is widely diffuse and allows the to transform a MIMO nonlinear system into a simple decoupled linear system, nevertheless relies on a precise modeling of the dynamic system and a perfect knowledge of dynamical parameters. For those reasons classical inverse dynamics controllers are affected by changes in dynamical system properties more advanced techniques able to adapt to system modifications like adaptive control have been developed.

Adaptive Control

Adaptive controller are different from other ordinary controller because the control parameters are time varying. Those parameters are adjusted by an online mechanism based on some signal of the closed-loop system. By means of this type of controller, it is possible to reach control goal even if the plant is subjected to uncertanties or modifications. There are many ways in which this controlled has been implemented, including adaptive computed-torque control, adaptive inertia-related control, adaptive control based on passivity, and adaptive control with desired compensation. Adaptive computed-torque control will be presented as an example. Failures in model parameters estimation for classical torque control can lead to mismatch terms that can be interpreted as nonlinear perturbation. We'll use the same approach of 3.5 but considering estimated parameters:

$$\tau = \hat{H}(q)(\ddot{q}_d + K_V \dot{e}_q + K_P e_q) + \hat{C}(q,\dot{q})\dot{q} + \hat{\tau_g}(q) \tag{3.7}$$

where $\hat{H}, \hat{C}, \hat{\tau}$ have the same functional form as H, C, τ_g. By means of the properties of the dynamical model to be linear wih respect to the system parameters it is possible to write:

$$hatH(q)\ddot{q} + \hat{C}(q,\dot{q}) + \hat{\tau_g}(q) = Y(q,\dot{q}\ddot{q})\hat{a} \tag{3.8}$$

where $Y(q,\dot{q}\ddot{q})$ is known as a regressor matrix and a is the vector of the estimated parameters. Applying the control law defined as 3.7 lead to:

$$\hat{H}(q)(\ddot{e}_q + K_V \dot{e}_q + K_P e_q) = Y(q,\dot{q}\ddot{q})\tilde{a} \tag{3.9}$$

where $\tilde{a} = \hat{a} - a$. Now under the assumption of $\ddot{q}$ measurable and non singularity of $\hat{H}(q)$ we rewrite the equation in state space form:

$$\begin{aligned} &\dot{x} = Ax + B\hat{H}^{-1}(q)Y(q,\dot{q}\ddot{q})\tilde{a} \\ &\text{with} \quad x = [e_q^T, \dot{e}_q^T]^T \\ &A = \begin{bmatrix} 0_n & I_n \\ -K_P & -K_V \end{bmatrix}, B = \begin{bmatrix} 0_n \\ I_n \end{bmatrix}, \end{aligned} \tag{3.10}$$

The new adaptive control law will then be:

$$\dot{\hat{a}} = -\Gamma^{-1} Y^T(q,\dot{q}\ddot{q})\hat{H}^{-1}(q)B^T P x \tag{3.11}$$

where Γ is a constant positive-definite matrix and P is a symmetric positive-definite constant matrix that satisfy:

$$PA + A^T P = -Q$$

Q is a symmetric positive definite matrix with coherent dimension. The stability of the closed loop system and the boundedness of internal signals can be assessed studing the Lyapunov stability of the function $\dot{V} = -x^T Q x$. For more details see [1]

Optimal Control

In place of determining the optimal sequence of decisions from the fixed state of the system, we wish to determine the optimal decision to be made at any state of the system. Only if we know the latter, do we understand the intrinsic structure of the solution.

Richard Bellman, Dynamic Programming, 1957. [Vinter, p. 435]

From a practical point of view, once the stability of a controller has been proved, nothing says that there is only one controller able to perform the control action. In other words since the stability proof doesn't determine a unique controller, generally it is possible to choose among different alternatives. It's pretty natural, in many contexts, that the choice of an optimal controller is preferred. The objective of optimal control theory, indeed, is to determine the control action that will cause a system to satisfy phisical constraints and at the same time to minimize (or maximize) some performance criterion. Generally the definition of this criterion (i.e. cost function) is done according with the application of the controller. Since there are many complex problems in control field that cannot be solved using classical techniques, optimal control theory has been studied and improved in 90's. At the same time, the increase in computation capability of calculators allows to implement those logics in many fields. However the design of an optimal controller generally relies on an exact model of the system. In fact, the presence of discrepancy between the model and the real system can lead to non-optimal solution that can easily end in an instable closed loop system. Let's consider a generic system described by nonlinear time-varying differential equation in $x \in R^n$:

$$\dot{x}(t) = f(x,t) + G(x,t)u \tag{3.12}$$

where $u \in R^m$ is the control input. We will consider the sistem without disturbancies term. We will refer as an example to a quadratic optimal control problem. Every optimal controller needs the definition of a cost function, as example:

$$z = H(x,t)x + K(x,t)u \tag{3.13}$$

such that: $H^T(x,t)K(x,t) = 0, K^T(x,t)K(x,t) = R(x,t) > 0$ and $H^T(x,t)H(x,t) = Q(x,t) > 0$. The quadratic cost function becomes:

$$\frac{1}{2}z^T z = \frac{1}{2}x^T Q(x,t)x + \frac{1}{2}u^T R(x,t)u$$

The optimal control law with the quadratic cost function can be derived from the solution of HJB (Hamilton-Jacobi-Bellman) equation for a positive-definite function $V(x,t)$ as in [2].

$$\begin{aligned} 0 = HJB = (x,t;V) = V_t(x,t) + V_x(x,t)f(x,t) \\ -\frac{1}{2}V_x(x,t)G(x,t)R^{-1}(x,t)G^T(x,t)V_x^T(x,t) + \frac{1}{2}Q(x,t) \end{aligned} \tag{3.14}$$

where $V_t = \frac{\partial V}{\partial t}$ and $V_x = \frac{\partial V}{\partial x^T}$. Then the inverse quadratic optimal control problem is to find a set $Q(x,t)$ and $R(x,t)$ for which $V(x,t)$ is a solution for the HJB. That defines the oprimal controller such as:

$$u = -R^{-1}(x,t)G^T(x,t)V_x^T(x,t) \tag{3.15}$$

Since in most application it is required to design a controller able to generate control inputs starting from observation of system outputs, there are three alternatives:

1. Design the controller in closed loop form and by means of a computational unit solve optimal control problem on-line
2. Precompute an optimal control law offline and then synthetize it into a special-purpouse digigtal controller.
3. Use a suboptimal controller whose parameters have been defined offline.

Anyway generally speaking the proper choice is made according to the application. The system evolution frequency for example gives many limitation to on-line solution of the optimal control problem. A typical implementation of optimal control theory is made in practice by means of Model Predictive Control.

Model Predictive Control

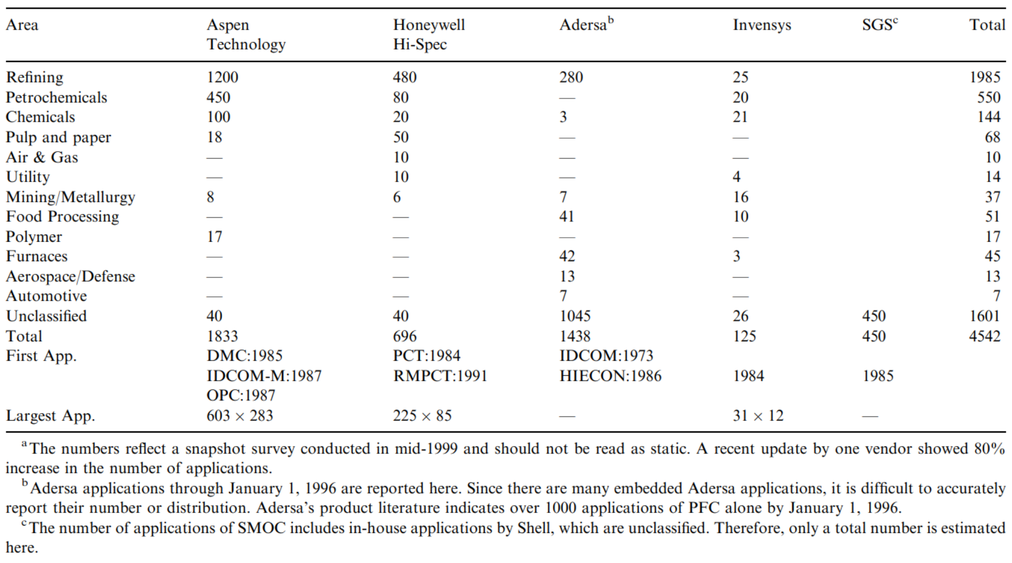
Discrete Dynamic Optimization Applied to On-line Optimal Control

Rafal, Stevens, AiChE Journal, 1968

The term Model Predictive Control (MPC from now on) designate a wide range of control strategies which use explicitly the model of the system to obtan the control action minimizing a specific cost function. The first development originated in the late seventies with IDCOM (Identification and Command) logic. Using impulse response for the plant model and optimiziang a quadratic objective function, This control, settled the basis of an architecture that has been developed considerably since then, expecially over the last two decades both within research community and industry. Such a succes is probably due to it's generality, in fact MPC formulation can be considered as one of the most general ways to pose a process control problem. It's formulation can integrate optimal control, stochastic control, control of processes with dead time and multivariable control. In general nonlinear processes which are frequently found in industry, can be handled by MPC logics. Futhermore , unless stability and robustness proof is difficult to asses because of the finite horizon, it has been found to be quite robust in many applications. Anyway, although both in industry and in research is widely diffuse, MPC has not yet reached the potential it suggest. One reason can be found in the fact that requires mathematical knowledge which is not usually available by control engineers in practice. This control architecture is generally an open framework that allows the development of many applications in use at the current time, not only in the industrial processes but also application to robotics, power plants, PVC plants, servos etc. Since MPC showed good performances in many of these applications as well as it's ability to operate whit few intervention for long periods of time, the interest in this control strategy is increasend in the last decades. This can be justified also by the increased capability in solving complex problem, sometimes complex rather than nonlinear. Many are the advantages with respect to other control techniques:

- Tuning is relatively easy.
- Implementing simple dynamics to more complex ones, it can be used to control a great variety of processes also including delay times, non-minimum phase or unstable systems.
- It can introduce feed forward control to compensate measurable disturbances.
- Easily defined for multivariable problems.
- Can be easily extended to treat the constraints and these can be systematically included during the design process.
- It is very useful when future references are known for example for robotic applications.
- It is an open methodology with many applications allowing further extensions.

Nevertheless it also has its drawbacks. First of all is that it's derivation is more complex than the one of classical PID controllers. Then, even if with a good model of the system we can prederivate the controller, the system can usually be subject to modification in it's dynamics, so the derivation needs to be done online, increasing significantly the computational effort. Moreover the introduction of constraints generally increase the computational time of the problem, that may result in a low frequency controller, not suitable for some applications. But computational time is not the only drawback. To be able to perform correctly, the system needs to be accurately modeled and that may not be possible in some cases in which for example the system is very complex or higly coupled. Even if there are many practical problems in implementing MPC for complex or high frequency control problems, it is a control logic suitable for many applications in which perform nicely.



Area	Aspen Technology	Honeywell Hi-Spec	Adersa[b]	Invensys	SGS[c]	Total
Refining	1200	480	280	25		1985
Petrochemicals	450	80	—	20		550
Chemicals	100	20	3	21		144
Pulp and paper	18	50	—	—		68
Air & Gas	—	10	—	—		10
Utility	—	10	—	4		14
Mining/Metallurgy	8	6	7	16		37
Food Processing	—	—	41	10		51
Polymer	17	—	—	—		17
Furnaces	—	—	42	3		45
Aerospace/Defense	—	—	13	—		13
Automotive	—	—	7	—		7
Unclassified	40	40	1045	26	450	1601
Total	1833	696	1438	125	450	4542
First App.	DMC:1985 IDCOM-M:1987 OPC:1987	PCT:1984 RMPCT:1991	IDCOM:1973 HIECON:1986	1984	1985	
Largest App.	603 × 283	225 × 85	—	31 × 12	—	

[a]The numbers reflect a snapshot survey conducted in mid-1999 and should not be read as static. A recent update by one vendor showed 80% increase in the number of applications.
[b]Adersa applications through January 1, 1996 are reported here. Since there are many embedded Adersa applications, it is difficult to accurately report their number or distribution. Adersa's product literature indicates over 1000 applications of PFC alone by January 1, 1996.
[c]The number of applications of SMOC includes in-house applications by Shell, which are unclassified. Therefore, only a total number is estimated here.

Table 3.1: Summary of linear MPC applications by areas

In [3] is presented an overview of commercially available model predictive control technology, both linear and nonlinear. The table 3.1 from that work shows the applications of MPC logic by area. Such a diffuse usage of this control logic collocate MPC as the second most used control methodology after PID. In the last decades MPC has been investigated in may fields as Process control (linear or nonlinear MPC), Automotive (Explicit, Hybrid MPC), Aerospace(LTV MPC), ICT(distributed/decentralized MPC), Energy or Finance (stochastic MPC). As most of the drawbacks are related to the model identification or optimization algorithms capability, fields in which research is working nowdays, MPC is still of high importance for research and industry. In this section basic concepts of Model Predictive Control will be presented in order to understand more in detail what we developed. More details in MPC theory can be found in [4]

Basic Concepts

As mentioned before Model Predictive Control use the model of the system and the definition a customized cost function in order to set and solve an optimization problem to find the correct control inputs to perform a specific control task. A diagram of how MPC basic architecture is schematized below:

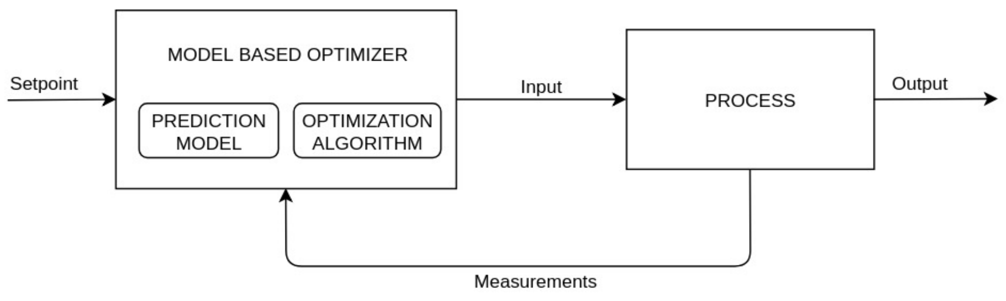


Figure 3.2: MPC basic architecture scheme

The model based optimizer (usually a computer) use optimization algorithms in accordance with the model of the system and given measures to minimize the cost function which is generally dependent on the error with respect to the given setpoint. Generally other constraints can be added to the problem in order for example to keep the states variables within feasible ranges. The output of the computation is then the input of the real process to be controlled. In practice the solution of the problem usually has two main limitation:

- Model of the process: the nature of the MPC is based on the possibility to forecast the future output of the system by means of a model. Because of that, to have a reliable prediction, we need the system model to be accurate.
- Optimization algoritm: as mentioned before, the more the system is complex and constraints are added to the problem, the more the problem will be computationally heavy. A proper optimization algoritm needs to be choose in accordance with the application of the controller.

Any controller belonging to MPC family is usually amenable to the following methodology:

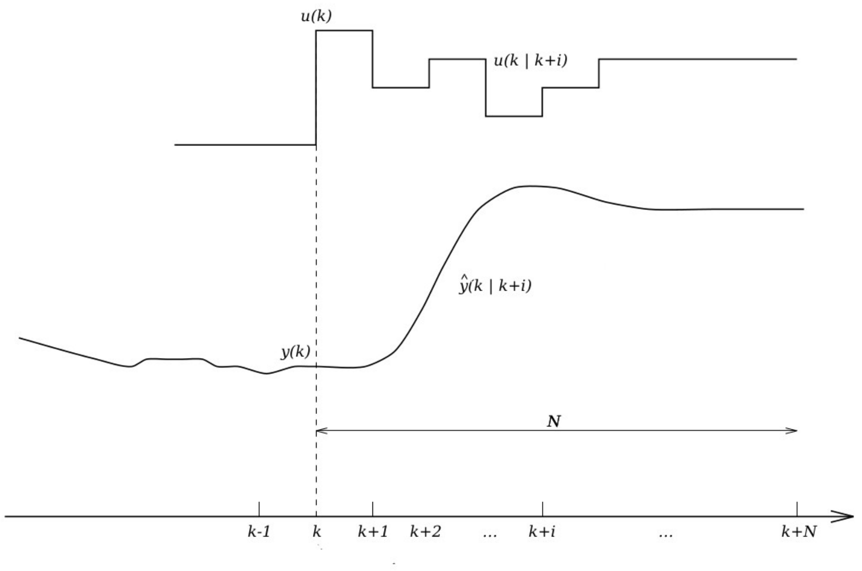


Figure 3.3: MPC strategy

1. Future outputs defined on a prediction horizon N as $\hat{y}(k|k+i)$ for $i = 1 \ldots N-1$ are predicted starting from past inputs and outputs as a function of the future control actions $u(k|k+i)$, $i = 0 \ldots N-1$.

2. The future control signal set is calculeted optimizing a given cost function to keep the predicted future output as close as possible to the given reference trajectory $y_d(k+i)$. The objective function is generally defined as a quadratic fuction with respect to this reference trajectory. Control effort term is also usually added to the cost in most cases. If the system is linear and the cost function is quadratic with respect to the error then an explicit solution can be found, otherwise merical methods have to be used instead.

3. The computed control action $u(k|k)$ is applied to the process then another optimization problem is set based on new measures in order to find $u(k+1|k+1)$ that will be in principle different from $u(k|k+1)$.

The sequence above repeats continuously following the so called Reciding Horizon principle. In order to be implemented, the model based optimizer, is designed following this scheme:

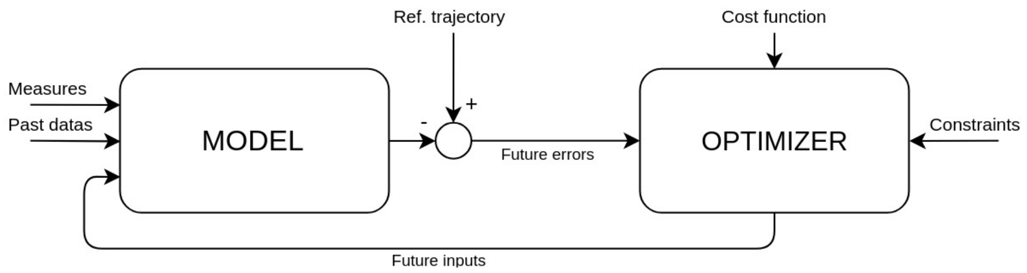


Figure 3.4: Model based optimizer

The definition of prediction model can be done in several ways like inpulse response, step response, transfer function, state space and more complex representation like neural networks for nonlinear systems. We will refer to the state space representation for the prediction model as it is easy to implement and to integrate.
Let's define the discrete linear time invariant (LTI) model of the process in state space form as:

$$\begin{cases} x_{k+1} &= Ax_k + Bu_k \\ y_{k+1} &= Cx_k \end{cases} \tag{3.16}$$

where $x \in R^{n_x}$ is the state, $y \in R^{n_y}$ and $u \in R^m$ are respectively the output and the control action and A, B and C are the matrices of the system. The prediction of the output for this model is given then by:

$$\hat{y}(k \mid k{+}i) = C\hat{x}(k \mid k{+}i) = C\left[A^i x(k) + \sum_{j=1}^{i} A^{j-1}Bu(k \mid k+i-j)\right] \tag{3.17}$$

which can be defined in matrix form as:

$$\begin{bmatrix} \hat{x}_k(1) \\ \hat{x}_k(2) \\ \vdots \\ \hat{x}_k(N) \end{bmatrix} = \underbrace{\begin{bmatrix} CB & 0 & \dots & 0 \\ CAB & CB & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{N-1}B & CA^{N-2}B & \dots & CB \end{bmatrix}}_{\bar{H}} \begin{bmatrix} u_k(0) \\ u_k(1) \\ \vdots \\ u_k(N) \end{bmatrix} + \underbrace{\begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix}}_{\bar{T}} \tag{3.18}$$

The definition of the cost function has a strong influence on stability proof as well as the convergence of numerical optimization methods. A preferrable choice is to consider a positive definite function of the error with respect to the reference trajecotry. As the problem can be easily moved a zero-reference problem we will refere to that case. For simplicity of notation we will refer to $\hat{y}(k \mid k+i)$ as $\hat{y}_{k|i}$

$$J(x_{k|0}, \mathbf{u}_k) = \sum_{i=0}^{N-1} \left[\hat{y}_{k|i}^T Q \hat{y}_{k|i} + u_{k|i}^T R u_{k|i}\right] + \hat{y}_{k|N}^T P \hat{y}_{k|N} \tag{3.19}$$

where $\mathbf{u} \in R^{mXN}$ with $\mathbf{u} = [u_{k|1} \; u_{k|2} \; ... \; u_{k|N-1}]$ is the control action that has to be found for the predicted horizon. R, Q and P are positive definite weight matrices. Let's write the cost function in matrix form:

$$J(x_{k|0}, \mathbf{u}) = x_{k|0}^T Q x_{k|0} + \begin{bmatrix} \hat{x}_{k|1} \\ x_{k|2} \\ \vdots \\ \hat{x}_{k|N} \end{bmatrix}^T \underbrace{\begin{bmatrix} Q & 0 & 0 & \dots & 0 \\ 0 & Q & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 \\ 0 & \dots & 0 & Q & 0 \\ 0 & 0 & \dots & 0 & Q \end{bmatrix}}_{\bar{Q}} \begin{bmatrix} \hat{x}_{k|1} \\ x_{k|2} \\ \vdots \\ \hat{x}_{k|N} \end{bmatrix} +$$

$$+ \begin{bmatrix} u_{k|0} \\ u_{k|1} \\ \vdots \\ u_{k|N-1} \end{bmatrix}^T \underbrace{\begin{bmatrix} R & 0 & \dots & 0 \\ 0 & R & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & R \end{bmatrix}}_{\bar{R}} \begin{bmatrix} u_{k|0} \\ u_{k|1} \\ \vdots \\ u_{k|N-1} \end{bmatrix}$$

Substituting the 3.17 we obtain:

$$\begin{aligned}
J(x_{k|0}, \mathbf{u}_k) &= x_{k|0}^T Q x_{k|0} + (\bar{S}\mathbf{u}_k + \bar{T}x_{k|0})^T \bar{Q}(\bar{S}\mathbf{u}_k + \bar{T}x_{k|0}) + \mathbf{u}_k^T \bar{R}\mathbf{u}_k \\
&= \frac{1}{2}\mathbf{u}_k^T \underbrace{2(\bar{R} + \bar{S}^T\bar{Q}\bar{S})}_{H} \mathbf{u}_k + x_{k|0}^T \underbrace{2\bar{T}^T\bar{Q}\bar{S}}_{F} \mathbf{u}_k + \frac{1}{2}x_{k|0} \underbrace{2(Q + \bar{T}^T\bar{Q}\bar{T})}_{Y} x_{k|0} \\
&= \frac{1}{2}\mathbf{u}_k^T H \mathbf{u}_k + x_{k|0} F \mathbf{u}_k + \frac{1}{2}x_{k|0} Y x_{k|0}
\end{aligned} \tag{3.20}$$

Once we have express the cost as a function of the measured initial state $x_k(0)$ and the future control action we can perform the optimization by zeroing the gradient:

$$\nabla_{\mathbf{u}_k} J(x_{k|0}, \mathbf{u}_k) = H\mathbf{u}_k + F^T x_{k|0} = 0 \tag{3.21}$$

$\mathbf{u}_k$ is then derived inverting the 3.5.1:

$$\mathbf{u}_k = \begin{bmatrix} u_{k|0} \\ u_{k|1} \\ u_{k|2} \\ \vdots \\ u_{k|N-1} \end{bmatrix} = -H^{-1}F^T x_{k|0}$$

The contorl law generated that can be derivated is then:

$$u(t) = -\left[\, I \ 0 \ \ldots \ 0 \,\right] H^{-1}Fx(t) \triangleq Kx(t) \tag{3.22}$$

The same result can be obtained from DARE (discrete-time Algebraic Riccati Equation) as done in [5]. By means of this result it is possible to implement the MPC with a simple feedback loop. Since it is possible, the derivation of 3.5.1 is done offline in this case, the online controller will then be fast given that the computational effort depend almost only on the inversion of H. Anyway the assuptions of Linear and time invariant system are considerably strong, many of practical systems present nonlinearities an time varying relationship. Because of that, different ways to minimize the 3.20 have to be found. Usually what is done in practice is to use algorithms to perform numerical optimization to find an approximation of the optimal solution. The main issue in the solution of the optimization problem is related to the well-definiteness of the problem. Indeed the correct choice of objective function form has to be done. usually the quadratic form allows to define convex problems in order to guarantee the unicity of the solution.

Constraints

In order to take into account feasibility sets and to gurantee stability, constraints are usually added to the optimization problem that becomes:

$$
\begin{aligned}
& min_{\mathbf{u}_k}\ J(x_{k|0}, \mathbf{u}_k) \\
\text{where } \ J(x_{k|0}, \mathbf{u}_k) &= \sum_{i=0}^{N-1} \left[\hat{y}_{k|i}^T Q \hat{y}_{k|i} + u_{k|i}^T R u_{k|i} \right] + \hat{y}_{k|N}^T P \hat{y}_{k|N} \\
\text{respecting } & \begin{cases} \hat{x}_{k+1} & = A\hat{x}_k + Bu_k \\ \hat{y}_{k+1} & = C\hat{x}_k \end{cases} \\
\text{s.t.} \qquad & 0 \leq f(x) \leq 0 \\
& 0 \leq g(u) \leq 0
\end{aligned}
\tag{3.23}
$$

This notation allows us to introduce both equality and inequality constraints. In paractice the constraints are divided into:

- **Input constraints**: generally represent philical limitation for the control $\mathbf{u}_k$ and are defined as "hard" constraints.
- **State/Output constraints**: usually come from rescrictions into the operational space, they can both be "soft" or "hard".

Hard constraints on the state or on the output gives generally complication in the implementation but soft constraints have to be defined properly according to the application. Moreover constraints allows a resonable control action to be generated when measured or estimated states moves outside the feasible sets. Given this outline, many are the possible constraints definition:

Band Constraints, Overshoot Constraints, Monotonic Behaviour, Nonminimum Phase Behaviour, Actuator Nonlinearities, Terminal State Equality Constraints, Terminal Set Constraints and more. Given that it is possible to define different constraints we will refer to terminal state equality constraint as it will be useful for the stability proof later on.
The terminal state constraint is basically defined as:

$$\hat{y}_{k|N} = C\left[A^i x_{k|0} + \sum_{j=1}^{i} A^{j-1} B u_{k|k+N-j}\right] = \bar{y}_N \tag{3.24}$$

that can be rewritten following the form in 3.23:

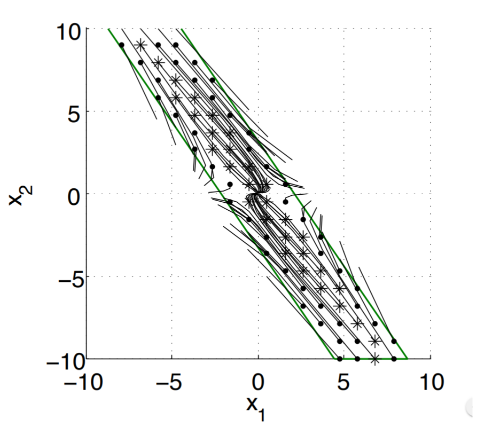
$$0 \leq \hat{y}_{k|N} - \bar{y}_N \leq 0 \tag{3.25}$$

Stability and Feasibility

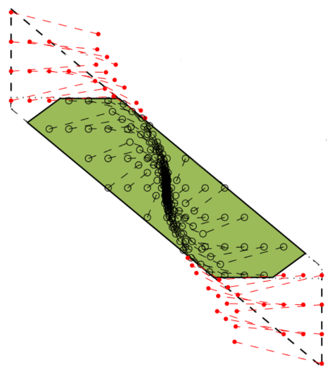
In this section stability and feasibility of MPC are investigated in order to give a clear understanding of what done to proof stability of the MPC we developed. To assess feasibility we need to investigate and proff is the problem have a solution and then to proof recursive feasibility, so the system accept a solution $\forall\ t > 0$. As well as feasibility, stability has to be proof in order to have convergence of the performed trajecotry. We will refer to a regulation problem, so the set point will be defined as the origin. The set X_k of initial states $x_{k|0}$ defined for any instant k that ensure feasibility for the MPC problem 3.23 is defined as:

$$\begin{aligned} X_k = \{\ x_{k|0} \in \mathbb{R}^n \mid \exists\ (u_{k|0}, \ldots, u_{k|N-1}) \text{ s.t. } x_{k|i} \in X, u_{k|i} \in U,& \\ \forall\ i = 0, \ldots, N-1,\ x_{k|N} \in X_f\ \}& \end{aligned} \tag{3.26}$$

where U is the region of the control action, X_f is the region of terminal states and $x_{k|i+1}$ is propagated following the 3.16 The feasible and unfeasible points in a regulation problem can be visualized on a graph (see 3.5(a)) and a region



(a) Feasibility points



(b) Feasibility region

Figure 3.5: Feasibility set

of feasible points can be defined (see 3.5(b))
Feasibility of a point does not guarantee the feasibility of the following inputs generated by the control action. Because of that Recursive feasibility has to be assessed. That means to show that:

$$\begin{gathered}\forall\ k\ \exists\ \mathbf{u}_k \in U \text{ s.t.} \\ \text{given } x_{k|0} \in X,\ x_{k|i} \in X\ \forall\ i = 1,\ \dots\ , N-1\end{gathered} \tag{3.27}$$

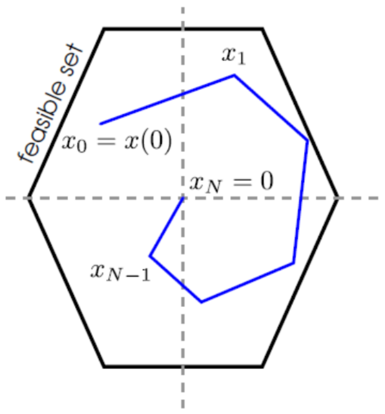


Figure 3.6: Recursive feasibility

While feasibility proof is straightforward because it is only affected by outputs hard constraints, stability is difficult to assess in some cases. Stability is a complex function of N, Q, R, P, U and admissible outputs of the system. Usually the stability proof of the MPC is verified imposing terminal constraints and weigths on the terminal state. There are many well studied stability proofs for MPC problems:

- Infinite horizon MPC (N = inf) with no additional constraints
- Terminal point constraints ($x_{k|N} = 0$)
- Relaxed terminal constraints ($x_{k|N} \in X_f$)

The equilibrium point of a controlled system is said to be Lyapunof stable in X if $\forall k \in \mathbb{N}$

$$\begin{gathered} \forall\ \epsilon > 0, \exists\ \delta(\epsilon) > 0 \text{ such that } \forall x_{k|0} \in X : \\ ||x_{k|0}|| \leq \delta(\epsilon) \implies ||x_{k|i}|| < \epsilon \forall\ i \in \mathbb{N} \end{gathered} \tag{3.28}$$

Stability proof is generally made by showing that the cost function to be minimized is a Lyapunov Function and admits an equilibrium point in the origin (regulation problem). The if J is a Lyapunov function, the equilibrium point is stable with region of attraction X.
A function $V : X \rightarrow\ \mathbb{R}_+$ is a Lyapunov function if $\forall\ x \in\ X$:

$$\begin{gathered} V(x) > 0 \rightarrow \forall\ x \neq 0 \\ V(0) = 0 \\ V(x_{k+1}) - V(x_k) \leq 0 \end{gathered} \tag{3.29}$$

In order to guarantee convergence, asymptotic stability in X of the equilibrium point has also to be verified. Hence it's needed to verify that it is Lyapunov stable and attractive in X ($\lim_{k\rightarrow\infty} ||x_k|| = 0\ \forall\ x\ \in\ X$). That can be verified by extending the 3.29 asking:

$$V(x_{k+1}) - V(x_k) < 0 \tag{3.30}$$

Now let's consider for simplicity the cas in which $C = I$ so the system in 3.16becomes:

$$x_{k+1} = Ax_k + Bu_k$$

We will assess stability by means of terminal constraints imposition (i.e. $x_{k|N} = 0$) considering the quadratic cost function in 3.19 with $J(x_{k|0}, \mathbf{u}_k) = J_k(x_0, \mathbf{u})$

$$J_k(x_0, \mathbf{u}) = \sum_{i=0}^{N-1} \underbrace{\left[x_{k|i}^T Q x_{k|i} + u_{k|i}^T R u_{k|i}\right]}_{q(x_i,u_i)} + \underbrace{x_{k|N}^T P x_{k|N}}_{p(x_N)} \tag{3.31}$$

$$\text{and} \quad x_n = 0$$

Recursive feasibility has first to be verified: Assuming the feasibility of the point x_0 and defining $[u_0^*,\ u_1^*,\ \dots,\ u_{N-1}^*]$ the optimal control sequence computed minimizing $J_0(x_0, \mathbf{u})$, following the MPC strategy we apply u_0^*. The evolution of the system will then be $x_1 = Ax_0 + Bu_0^*$. Now at instant $k = 1$ the control sequence $[u_0^*,\ u_1^*,\ \dots,\ u_{N-1}^*,\ 0]$ is feasible because applying a null control input and given that $x_N = 0$ the resulting final state is $x_N + 1 = 0$. This implies that, in presence of end point constraints, the recursive feasibility is verified assuming x_0 feasible.
Once feasibility has been assessed, stability has to be investigated by checking if J is a Lyapunov function. So, since we are interested in asymptotic stability, we need to verify;

$$J_0^*(x_1) < J_0^*(x_0) \quad \forall x_0 \neq 0 \tag{3.32}$$

Follows that:

$$
\begin{aligned}
J_0^*(x_0) &= \underbrace{\cancel{p(x_N)}}_{=0} + \sum_{i=0}^{N-1} q(x_i, u_i^*) \\
J_0^*(x_1) &\leq \tilde{J}_0(x_1) = \sum_{i=1}^{N} q(x_i, u_i^*) \\
= \sum_{i=1}^{N-1} q(x_1, u_i^*) &- q(x_0, u_0^*) + q(x_N, u_N) \\
&= \ J_0^*(x_0) - q(x_0, u_0^*) + \underbrace{\cancel{q(0,0)}}_{=0}
\end{aligned}
\tag{3.33}
$$

Then what can be said is that:

$$
J_0^*(x_1) - J_0^*(x_0) \leq \tilde{J}_0^*(x_1) - J_0^*(x_0) \leq -q(x_0, u_0^*) \leq 0 \tag{3.34}
$$

Hence, $J * (x)$ is a Lyapunov function, so the point $x_N = 0$ is an asymptotically stable point for the control law. This concept can be generalized for different kind of problem. Anyway, as mentioned before, there are other ways to verify stability which depends on how is the problem defined. Summarizing, terminal constraints imposition provides a sufficient condition for stability in practice what is done is to enlarge the control horizon and sampling since the system perform stability.

NMPC

LTV MPC

Chapter 4

Mobile Manipulators Control Problem

We have seen that Mobile Manipulators are systems where locomotion and manipulation tasks are held simultaneously. The coordination of this two tasks is not trivial since the mobile platform combined with the manipulator is a redundant system. So, the same point in workspace can be reached moving the mobile base, or moving the manipulator, or with a combined motion of the two. In addition to this kinematic issue, also the dynamic interaction between mobile base and manipulator should be taken into account in a control logic.

In literature the control of mobile manipulators mainly followed two strategies. The first approach is the separate control of the mobile base and of the manipulator arm, considering then their interaction. With this approach is possible to simply solve the kinematic redundancy of the system. A reference for this approach can be [6] or [7]. The second approach is the coordination of the motion of the base and the manipulator through the control of the whole integrated redundant system. We will split this chapter in the description of the control of the mobile base, followed in a seceond section by the problem of the integration of the manipulator in the control.

Control of Mobile Robots

Facing the control of the mobile base it is needed to deal with a nonholonomic system, which is nonlinear, as it has been written before. Usually for the control of the motion of the mobile base the kinematic model is used applying as inputs to the system the commands for the longitudinal velocity v and the steering velocity ω. In this way it is possible to control directly the generalized velocities of the base, i.e. $\dot{x}, \dot{y}, \dot{\theta}$. It would be certainly possible to use a dynamic based control of the motion through torques commands to the wheels of the base, but in most mobile robots, in particular in the one we used, the dynamic perturbation is very small, or it can be canceled through a proper state feedback. Moreover it is possible to command the mobile robot only via velocity commands. The control of the mobile base can be implemented in two ways: regulation, i.e. driving the robot to a constant position and/or orientation, and trajectory tracking, i.e. forcing the robot to follow a time-varying reference trajectory.

Regulation

The regulation problem can be addressed in two ways.

Cartesian Regulation

The robot (a unicycle) has to reach a point whose coordinates (x_{ref}, y_{ref}) are defined in the Cartesian space. In this problem there is no need to control also the orientation of the robot. In literature several control laws have been proposed. Among others [8] reports: defining the Cartesian error $\mathbf{e}_P = (x_{ref} - x, y_{ref} - y) = (e_{P_x}, e_{P_y})$ the control inputs can be obtained as

$$v = k_1(e_{P_x} \cos\theta + e_{P_y} \sin\theta) \tag{4.1}$$

$$\omega = k_2 \left(\text{Atan2}\left(e_{P_y}, e_{P_x}\right) - \theta - \pi\right) \tag{4.2}$$

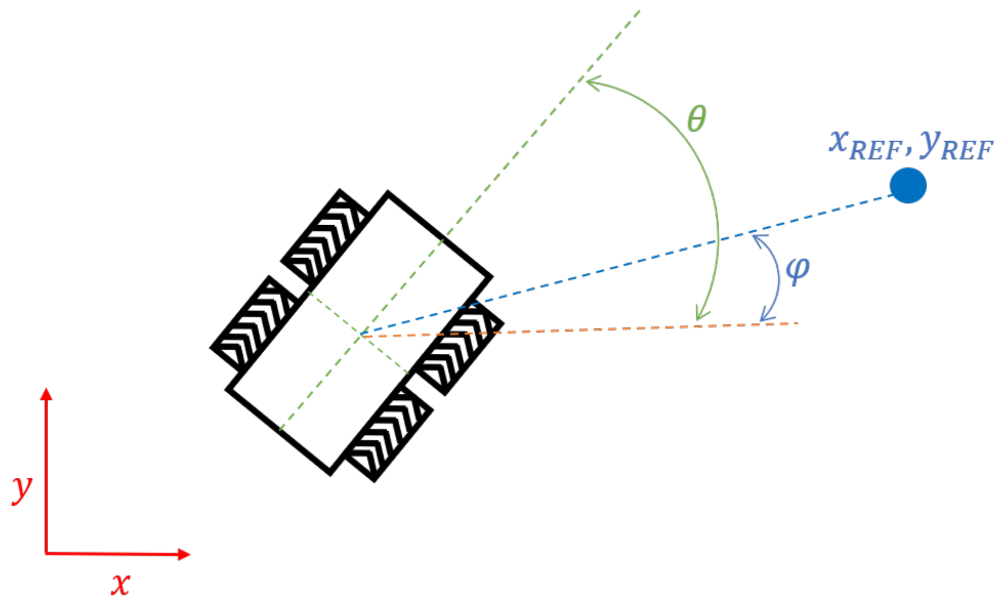


Figure 4.1: Cartesian Regulation problem

where k_1 and k_2 are positive constants. As [8] too says, these commands can be directly interpreted, because v is proportional to the Cartesian error, driving the robot to $\mathbf{e}_P = 0$, while the ω is steering the robot in the direction pointing to (x_{ref}, y_{ref}), i.e. ω is proportional to the difference $\theta - \varphi$, where φ is Atan2 $\left(e_{P_y}, e_{P_x}\right)$.

Posture Regulation

The robot has to get to a certain configuration, i.e. Cartesian position and orientation. Also in this case, the literature is full of control laws for this kind of problem. We report the control law found in [9] and in [10]. Defining, for a practical advantage, the transformation:

$$\mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{ref} - x \\ y_{ref} - y \\ \theta_{ref} - \theta \end{bmatrix} \tag{4.3}$$

Which are the errors in a local frame of the mobile robot, where e_1 is the error in the longitudinal direction and e_2 is the error in the transversal direction.

Deriving the above equation taking into account the kinematic relation 2.1:

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} -v + \omega e_2 \\ -\omega e_1 \\ \omega \end{bmatrix} \tag{4.4}$$

The proposed control law for v and ω is:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} -k_1 e_1 \\ -k_2 e_3 + e_2^2 \sin(t) \end{bmatrix} \tag{4.5}$$

where k_1 and k_2 are positive constants. So the closed-loop dynamics becomes:

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} k_1 e_1 + \omega e_2 \\ -\omega e_1 \\ -k_2 e_3 + e_2^2 \sin(t) \end{bmatrix} \tag{4.6}$$

For the stability proof of this control law we refer to [9].

Trajectory Tracking

A *trajectory* is a path, i.e. the locus of points which the robot has to follow in the execution of the assigned motion, over which a timing law has been assigned.
So a trajectory can be represented as the time history $x_d(t), y_d(t)$, but in the case of a mobile robot the trajectory has also to be admissible, because it has to respect the nonholonomy of the system, i.e. it must satisfy the relation 2.36:

$$\begin{aligned} \dot{x}_d &= v_d \cos\theta_d \\ \dot{y}_d &= v_d \sin\theta_d \\ \dot{\theta}_d &= \omega_d \end{aligned} \tag{4.7}$$

Usually reference trajectories for mobile robots are being computed by planning algorithms so that 4.7 is satisfied, but they give in output just the

Cartesian coordinates' time history. The orientation reference trajecory can be computed as:

$$\theta_d(t) = \text{Atan2}(\dot{x}_d(t), \dot{x}_d(t)) \tag{4.8}$$

And also the resulting reference control inputs can be calculated.

$$v_d(t) = \sqrt{\dot{x}_d^2(t) + \dot{y}_d^2(t)} \tag{4.9}$$

$$\omega_d(t) = \frac{\ddot{y}_d(t)\dot{x}_d(t) - \ddot{x}_d(t)\dot{y}_d(t)}{\dot{x}_d^2(t) + \dot{y}_d^2(t)} \tag{4.10}$$

To quantify the tracking control objective the tracking error vector **e** is defined as before (4.3), but in trajectory tracking problems the derivative of **e** is different than in regulation problems because the derivative of the reference trajectory is no more null.

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} v_{ref} \cos e_3 - v + \omega e_2 \\ v_{ref} \sin e_3 - \omega e_1 \\ \omega_{ref} - \omega \end{bmatrix} \tag{4.11}$$

The tracking error dynamics is nonlinear, so the control of the system can be addressed with a nonlinear control or linearizing the system with a suitable transformation.

Nonlinear Control

A common used Nonlinear Control for trajectory tracking can be found in [11],[10],[9] and [8], as follows:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} -k_1 e_1 + v_{ref} \cos e_3 \\ -v_{ref} \frac{\sin e_3}{e_3} - k_2 e_3 + \omega_{ref} \end{bmatrix} \tag{4.12}$$

Where k_2 is a positive constant and k_1 and k_3 are positive gains which depend on the values of v_{ref} and ω_{ref}.
So the closed-loop error dynamics becomes:

$$\begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \end{bmatrix} = \begin{bmatrix} \omega e_2 - k_1 e_1 \\ v_{ref} \sin e_3 - \omega e_1 \\ -v_{ref} \frac{\sin e_3}{e_3} - k_2 e_3 \end{bmatrix} \tag{4.13}$$

Also here, for the stability proof of this control law we refer to [9].

Feedback Linearization

Input/Output linearization is a commonly used way to face both regulation and trajectory tracking problems for mobile robots, but it has been seen that paradoxically it solves more easily trajectory tracking problems than regulation ones. The idea of Feedback Linearization is to move the control problem to the trajectory of a point P placed at a distance ϵ from the centre of the mobile robot in the direction of the longitudinal velocity. Its coordinates are

$$\begin{aligned} x_P &= x + \epsilon \cos\theta \\ y_P &= y + \epsilon \sin\theta \end{aligned} \tag{4.14}$$

Which differentiated are the velocities of P:

$$\begin{bmatrix} \dot{x}_P \\ \dot{y}_P \end{bmatrix} = \begin{bmatrix} \dot{x}_P - \dot{\theta}\epsilon \sin\theta \\ \dot{y}_P + \dot{\theta}\epsilon \cos\theta \end{bmatrix} = \begin{bmatrix} \cos\theta & -\epsilon \sin\theta \\ \sin\theta & \epsilon \cos\theta \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} = T(\theta) \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{4.15}$$

The determinant of matrix $T(\theta)$ is equal to ϵ and therefore always different from 0 as long as $b = 0$. Thence the inverted relation is:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\frac{1}{\epsilon}\sin\theta & \frac{1}{\epsilon}\cos\theta \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{4.16}$$

Where u_1 and u_2 are the new linearized output of the controller. Indeed thanks to the feedback linearization, a simple linear model can be rewritten:

$$\begin{bmatrix} \dot{x}_P \\ \dot{y}_P \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{4.17}$$

And so a very simple linear controller can be defined as in [8] without using a state error, but an output error:

$$\begin{aligned} u_1 &= \dot{x}_{P_{ref}} + k_1 \left(x_{Pref} - x_P \right) \\ u_2 &= \dot{y}_{P_{ref}} + k_2 \left(y_{Pref} - y_P \right) \end{aligned} \tag{4.18}$$

Where k_1 and k_2 are positive gains.
It has to be noted that the orientation of the unicycle θ is not controlled, but its time evolution due to the control inputs u_1 and u_2 can be modeled as:

$$\dot{\theta} = \frac{u_2 \cos\theta - u_1 \sin\theta}{\epsilon} \tag{4.19}$$

Also a notation on the value of ϵ has to be added, because with smaller values of ϵ is possible to track more precisely a trajectory even if it involves curves with a small radius, but decreasing ϵ, $T(\theta)$ tends to be singular, therefore leading to much higher values of v and ω.

Control of Mobile Manipulators

Although Mobile Manipulators are a known technology from decades, the control of these systems is still sparse. In literature many approaches have been proposed to face the kinematic redundancy of the Mobile Manipulator system.
A typical Mobile Manipulator consists of a nonholonomic base with 2 degrees of freedom (considering the nonholonomic constraint on the generalized velocities) and an arm which usually have 6 degrees of freedom, for a total of $n = 8$; anyway, for vector X_{ee} of the position and orientation of the end effector position has only $m = 6$ components to be defined. The remaining $r = n - m = 2$ redundant degrees of freedom can be exploited to solve the kinematic redundancy problem, mentioned in section 2.2.1, satisfying a set of user-defined additional tasks. As explained in [12],[13],[14] and [15], these additional tasks are described by a set of internal coordinates which defined as kinematic function of the coordinates $\xi = \left[x^T; q^T\right]$

$$\Phi = g(\xi) \tag{4.20}$$

These internal coordinates can have different physical meanings (like the distance between the wrist of the manipulator from the arm) or mathematical meanings (such as the projection of the gradient of an objecive function). There is not a fixed set of kinematic functions to pick, but the user can define those according to the task that has to be performed. A new extended vector X_{ext} can be defined as:

$$X_{ext} = \begin{bmatrix} X_{ee} \\ \Phi \end{bmatrix} \tag{4.21}$$

The control problem now becomes a trajectory tracking of $X_{extref}(t) = \left[X_{ee}{}_{ref}^{T}(t) \; ; \; \Phi_{ref}^{T}(t) \right]$, where $\Phi_{ref}(t)$ is defined by the relation 4.20 over time. The trajectory tracking problem usually consists in the optimization of an objective function $\varphi(q)$ defined with one or multiple properly weighted criteria, some of which are shown in [16]. The most used optimization criteria are:

- The maximization of the manipulability of the system. See [17]. A short description of what manipulability will follow.
- The minimization of the torque inputs. See [18].
- The minimization of the deviation from the mid-range joints of the manipulator. See [19].

Clearly in addition to all these terms one is added for the minimization of the tracking error for all the vector X_{ext}.
It is not mandatory to extend the controlled vector adding internal coordinates in order to solve the kinematic redundancy problem. The additional tasks that the robot is asked to accomplish can be obtained adding a term to be minimized, e.g. the manipulability index, in the optimization problem. See for example [20] or [21] where the control is divided in two terms, one term that imposes the joint velocities through the pseudo-inverse matrix of $J(q)$ and one term to maximize the manipulability of the system.
Exposing these methods we haven't had to specify wheter a kinematic or dynamic model is used, leaving the user the possibility to choose the more suitable one with respect to the type of task the robot has to perform. If, for example, the planned motions of the robot can involve an high dynamic influence of the base on the arm and/or viceversa, a dynamic model should be

considered, using as input to the system the joint torques. A modelisation of the compensation of the dynamic effects between arm and base is discussed in [22]. Moreover, in the last years also some adaptive and robust controllers are being developed to deal with dynamic uncertainties, but they will not be discussed in this work. We refer to [23] for a fairly thorough discussion on the subject.

Manipulability

Kinematically the manipulability of a manipulator is a measure of its attitude to arbitrarily change the end effector position and orientation.
A common manipulability index has been presented in [24] [25].

$$w(q) = \sqrt{\det\left(J(q)J^T(q)\right)} \tag{4.22}$$

Where $J(q)$ is the Jacobian matrix that links the joints velocities to the end effector velocities. It is easy to see that the above expression is always positive and it reduces to $w(q) = |\det\left(J(q)\right)|$ when $J(q)$ is square.
When $w(q) = 1$ the end effector can move isotropically in all directions in the operational space.
Both the mobile platform and the manipulator have contributions on the manipulability: if $J(q) = \begin{bmatrix} J_b(q) & J_a(q) \end{bmatrix}$ then

$$w(q) = \sqrt{\det\left(J(q)J^T(q)\right)} = \sqrt{\det\left(J_b(q)J_b^T(q)\right) + \det\left(J_a(q)J_a^T(q)\right)} \tag{4.23}$$

It is possible to see that the base degrees of freedom increase the manipulability of the manipulator, i.e. even if the manipulator is in a singular configuration, $w(q) \neq 0$. However, in certain cases w can be equal to 0.

Obstacle Avoidance for Mobile Manipulators

Usually the main task assigned to a Mobile Manipulator is to follow a pre-planned trajectory in the Cartesian space with the end effector. However,

the robot doesn't know the environment in which it is. The robot motion has to be controlled such that a minimum distance between the robot links and the obstacles which can be present in the environment is kept greater than a security margin.
The pioneering work of [26] exploits the concept of *artificial potential fields* in obstacle avoidance within the control of manipulators with a fixed base, or of mobile robots. A potential field is generated in the Cartesian space decreasing towards the goal position of the robot and increasin going in proximity of obstacles. An example of the expression of the potential field can be:

$$\mathcal{U} = \mathcal{U}_{target} + \mathcal{U}_{obstacle} \tag{4.24}$$

$$\begin{aligned} \mathcal{U}_{target} &= \frac{1}{2}k\left(x - x_D\right)^2 \\ \mathcal{U}_{obstacle} &= \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho} - \frac{1}{\rho_0}\right)^2 & \text{if } \rho \leq \rho_0 \\ 0 & \text{if } \rho > \rho_0 \end{cases} \end{aligned} \tag{4.25}$$

Where $k$andη are positive constant, ρ is a measure of the distance between a point of the robot and the obstacle and ρ_0 is a security distance margin from the obstacle. Given a point in the Cartesian space a virtual force is

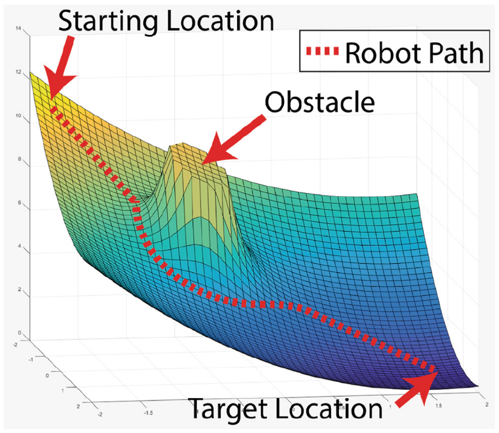


Figure 4.2: Example of artificial potential field

generated by the potential field in the direction of the field gradient.

$$\begin{aligned} F^*_{target} &= -grad\left[\mathcal{U}_{\sqcup\dashv\nabla\}]\sqcup}\right] = -k\,(x - x_D)^2 \\ F^*_{obstacle} &= -grad\left[\mathcal{U}_{\wr\lfloor\int\sqcup\dashv\rfloor\updownarrow]}\right] = \begin{cases} \eta\left(\frac{1}{\rho} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2}\frac{\partial\rho}{\partial x} & \text{if } \rho \leq \rho_0 \\ 0 & \text{if } \rho > \rho_0 \end{cases} \end{aligned} \tag{4.26}$$

Multiple points can be chosen on the manipulator arm, to avoid collisions with all the links of the robot. The forces generated on all of these points are then projected on joint torques space through the Jacobian matrices. This way the manipulator arm, or the mobile robot, is induced by the joint torques to move away from the obstacles.
The system subjected to the artificial potential field is stable, but with approach it is very easy to run into local minima where the system is stable, but the robot is not in the target configuration.
With systems like Mobile Manipulators the obstacle avoidance problem can also be formulated as an optimization problem. Exploiting its increased number of joints a Mobile Manipulator can follow a primary task, i.e. trajectory traking of the end effector, while simoultaneously optimizing a secondary objective, and/or satisfying certain constraints by utilisation of the system's redundancy. In this sense, obstacle avoidance naturally arises as a secondary objective to be achieved, as explained in [27],[28] or in [29], where an obstacle avoidance term is added in the objective function of the optimization problem for the control of Mobile Manipulators

$$\varphi_{OA}(q) = \sum_{i=1}^{l} \alpha_i p(\rho_i(q)) \tag{4.27}$$

Where l is the number of points where the obstacle avoidance is considered, α is a weighting factor and p is a penalty function depending on the distance of the point from the obstacle.
Another approach on Mobile Manipualators obstacle avoidance that deserves to be mentioned is the Elastic Strips method proposed in [30] where the motion of the robot is chosen between a set of homotopic collision-free paths through a potential field-based control algorithm.
A general issue that each approach has to deal with is the calculation of the

minimum distance ρ to an obstacle, and consequently the geometric modelling of the obstacles. More recent obstacle avoidance strategies ([31]) have overcome the complex 3D modelisation of the environment or of the robot. The approach is based on the exploitation of perception data available only from simple proximity sensors distributed on the robot, in order to deviate pre-planned motions exploiting the robot redundancy while executing the primary task. However, this strategy does not guarantee to complete the primary task in any environment.

Grasping

....

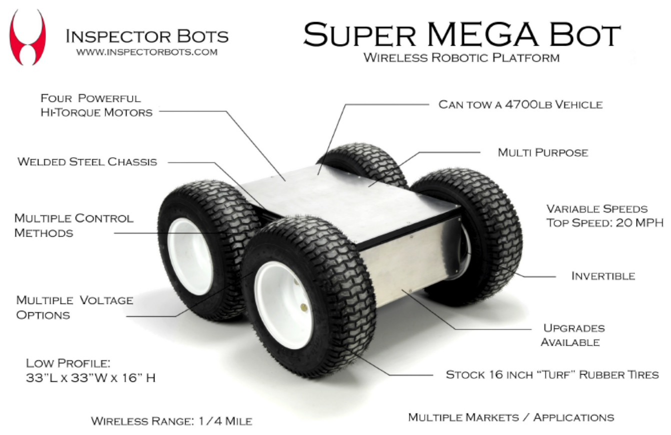
Chapter 5

Controller

Chapter 6

Experiments Setup

Super-Mega-Bot Mobile Base



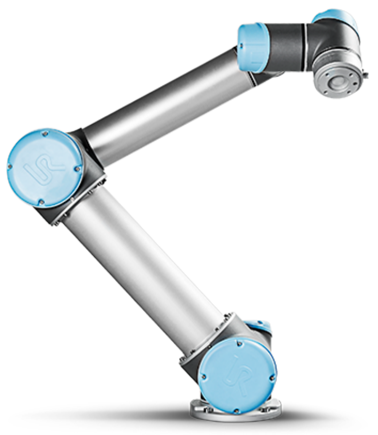
The InspectorBots Super Mega Bot robot platform is a Heavy-Duty, All-Terrain, 4WD Robotic Platform. It is a very rugged, Indoor/outdoor, remotely operated platform and can be fitted with a variety of sensors, cameras and equipment. The MEGA Bot has been designed to be Modular and Reconfigurable, so an end user can swap out various modules. There are several methods available to control the SMB including: RC radio, Analog Joystick, Wireless Modem, or Microcomputer.
Its base model comes with:

- Chassis: Powder Coated Steel and Aluminum
- Dimensions: 84cm L x 84cm W x 40.64cm H, Weight: 109 kg
- Drive: Four Electric Motors
- Load Capacity: 113,5 kg
- Speed Controllers: RoboteQ 2x VDC2450
- Speed: 0-16 km/h
- Suspension: Rigid
- Tires: Interchangeable, Pneumatic, 40.64cm Diameter Turf Tires

`https://www.youtube.com/watch?v=3X8IWnR1QDI`

UR5 Manipulator

The UR5 is a robot developed by Universal Robots, a Danish company. According to the company the key benefits of their robots are that they are light weight, safe and easy to use. The robotic arm is regarded as safe, because it will stop acting as soon as the robot hits an object sensed by a force sensor in one of the joints. UR5 is ideal for automating low-weight processing tasks like picking, placing and testing. The medium-sized robot arm is easy to program, fast to set up and, always according to UR, offers one of the fastest payback times in the industry.
In the following table the specifications given by Universal Robots are stated.



ADDAMS

SCRITTA MALISSIMO: The goal of this mobile manipulator is to fill the gap between small and large production. Indeed small production companies could not have the need to have a static robot, still needing an automatized process. Maybe they also don't have the money for the sensors needed to accomplish things. So there is the need of a robot able to substitute a person in the process but still having a user interface.

Performance		
Repeatibility	± 0.1 mm	
Ambient temperature range	0-50°	
Power Consumption	Min 90W, Typical 150W, Max 325W	
Collaboration operation	15 advanced adjustable safety functions	
Specifications		
Payload	5 kg	
Reach	850 mm	
Degrees of freedom	6 rotating joints	
Programming	Polyscope graphical user interface on 12 inch touchscreen	
Movement		
	Working range	Maximum speed
Base	± 360°	± 180°/sec
Shoulder	± 360°	± 180°/sec
Wrist1	± 360°	± 180°/sec
Wrist2	± 360°	± 180°/sec
Wrist3	± 360°	± 180°/sec
Typical tool		1m/sec
Physical		
Footprint	149mm	
Materials	Aluminium, PP Plastics	
Weight (with cable)	18.4 kg	

Mission

Sensors

MPC

Kinematic model based

Dynamic model based

Chapter 7

Results

Chapter 8

Conclusions

future: bisgonerebbe capire quale è il vanytaggio di un sistema non lineare rispetto a uno linearizzato

List of Figures

List of Tables

A

References

[1] J. J. Craig, P. Hsu, and S. S. Sastry, "Adaptive control of mechanical manipulators," *The International Journal of Robotics Research*, vol. 6, no. 2, pp. 16–28, 1987.

[2] D. E. Kirk, *Optimal control theory: an introduction.* Springer, 1970.

[3] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control engineering practice*, vol. 11, no. 7, pp. 733–764, 2003.

[4] E. F. Camacho and C. B. Alba, *Model predictive control.* Springer Science & Business Media, 2013.

[5] L. Magni and R. Scattolini, *Complementi di controlli automatici.* Pitagora, 2006.

[6] K. Liu and F. L. Lewis, "Decentralized continuous robust controller for mobile robots," in *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pp. 1822–1827, IEEE, 1990.

[7] J. H. Chung, S. A. Velinsky, and R. A. Hess, "Interaction control of a redundant mobile manipulator," *The International Journal of Robotics Research*, vol. 17, no. 12, pp. 1302–1309, 1998.

[8] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: modelling, planning and control.* Springer Science & Business Media, 2010.

[9] W. Dixon, D. M. Dawson, E. Zergeroglu, and A. Behal, *Nonlinear control of wheeled mobile robots.* Springer, 2001.

[10] C. Samson, "Control of chained systems application to path following and time-varying point-stabilization of mobile robots," *IEEE transactions on Automatic Control*, vol. 40, no. 1, pp. 64–77, 1995.

[11] C. C. d. Wit, H. Khennouf, C. Samson, and O. J. Sordalen, "Nonlinear control design for mobile robots," in *Recent trends in mobile robots*, pp. 121–156, World Scientific, 1993.

[12] B. Bayle, J.-Y. Fourquet, F. Lamiraux, and M. Renaud, "Kinematic control of wheeled mobile manipulators.," in *IROS*, pp. 1572–1577, 2002.

[13] H. Seraji, "A unified approach to motion control of mobile manipulators," *The International Journal of Robotics Research*, vol. 17, no. 2, pp. 107–118, 1998.

[14] H. Seraji, "An on-line approach to coordinated mobility and manipulation," in *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pp. 28–35, IEEE, 1993.

[15] W. Miksch and D. Schroeder, "Performance-functional based controller design for a mobile manipulator," in *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pp. 227–232, IEEE, 1992.

[16] F. G. Pin and J.-C. Culioli, "Multi-criteria position and configuration optimization for redundant platform/manipulator systems," in *Intelligent Robots and Systems' 90.'Towards a New Frontier of Applications', Proceedings. IROS'90. IEEE International Workshop on*, pp. 103–107, IEEE, 1990.

[17] Y. Yamamoto and X. Yun, "Coordinating locomotion and manipulation of a mobile manipulator," in *Decision and Control, 1992., Proceedings of the 31st IEEE Conference on*, pp. 2643–2648, IEEE, 1992.

[18] M. Chen and A. Zalzala, "Dynamic modelling and genetic-based trajectory generation for non-holonomic mobile manipulators," *Control Engineering Practice*, vol. 5, no. 1, pp. 39–48, 1997.

[19] O. Khatib, “Mobile manipulation: The robotic assistant,” *Robotics and Autonomous Systems*, vol. 26, no. 2-3, pp. 175–183, 1999.

[20] B. Bayle, J.-Y. Fourquet, and M. Renaud, “A coordination strategy for mobile manipulation,” in *Proceedings of 6th Intern. Conference on Intelligent Autonomous Systems (IAS-6)*, Citeseer, 2000.

[21] B. Bayle, J.-Y. Fourquet, and M. Renaud, “Manipulability of wheeled mobile manipulators: Application to motion generation,” *The International Journal of Robotics Research*, vol. 22, no. 7-8, pp. 565–581, 2003.

[22] Y. Yamamoto and X. Yun, “Effect of the dynamic interaction on coordinated control of mobile manipulators,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 5, pp. 816–824, 1996.

[23] Z. Li and S. S. Ge, *Fundamentals in modeling and control of mobile manipulators.* CRC Press, 2016.

[24] T. Yoshikawa, “Analysis and control of robot manipulators with redundancy,” in *Robotics research: the first international symposium*, pp. 735–747, MIT press Cambridge, MA, 1983.

[25] T. Yoshikawa, “Manipulability of robotic mechanisms,” *The international journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[26] O. Khatib, “Real-time obstacle avoidance for manipulators and mobile robots,” in *Autonomous robot vehicles*, pp. 396–404, Springer, 1986.

[27] Y. Yamamoto and X. Yun, “Coordinated obstacle avoidance of a mobile manipulator,” in *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, vol. 3, pp. 2255–2260, IEEE, 1995.

[28] H. G. Tanner and K. J. Kyriakopoulos, “Nonholonomic motion planning for mobile manipulators,” in *IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1233–1238, IEEE; 1999, 2000.

[29] V. Perdereau, C. Passi, and M. Drouin, “Real-time control of redundant robotic manipulators for mobile obstacle avoidance,” *Robotics and Autonomous Systems*, vol. 41, no. 1, pp. 41–59, 2002.

[30] O. Brock, O. Khatib, and S. Viji, “Task-consistent obstacle avoidance and motion behavior for mobile manipulation,” in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 1, pp. 388–393, IEEE, 2002.

[31] P. Falco and C. Natale, “Low-level flexible planning for mobile manipulators: a distributed perception approach,” *Advanced Robotics*, vol. 28, no. 21, pp. 1431–1444, 2014.