Parallel Robots: Mechanics and Control

Dehaeze Thomas

Table of Contents

1 Introduction

This book is intended to give some analysis and design tools for the increase number of engineers and researchers who are interested in the design and implementation of parallel robots. A systematic approach is presented to analyze the kinematics, dynamics and control of parallel robots. To define the motion characteristics of such robots, it is necessary to represent 3D motion of the robot moving platform with respect to a fixed coordinate. This issue leads to the requirements for 3D representation of position, orientation and motion of bodies in space. In chapter 2, such representation are introduced with emphasis on screw coordinates, which makes the representation of the general motion of the robot much easier to follow.

Kinematic analysis refers to the study of robot motion geometry without considering the forces and torques that cause the motion. In this analysis (chapter 3), the relation between the geometrical parameters of the manipulator and the final motion of the moving platform is derived and analyzed.

In Chapter 4, the kinematics analysis of robot manipulators is further examined beyond static positioning. Jacobian analysis not only reveals the relation between the joint variable velocities of a parallel manipulator and the moving platform linear and angular velocities, but also constructs the transformation needed to find the actuator forces from the forces and moments acting on the moving platform. A systematic means to perform Jacobian analysis of parallel manipulators is given in this chapter.

Dynamic analysis of parallel manipulators presents an inherent complexity due to their closed-loop structure and kinematic constraints. Nevertheless, dynamic modeling is quite important for the control, in particular because parallel manipulators are preferred in applications where precise positioning and suitable dynamic performance under high loads are the prime requirements. In Chapter 5, the dynamic analysis of such robots is examined through three methods, namely the Newton-Euler principle of virtual work and Lagrange formations. Furthermore, a method is presented in this chapter to formulate the dynamic equation of parallel robots into closed form, by which the dynamic matrices are more tractable, and dynamics verification becomes feasible.

The control of parallel robots is elaborated in the last two chapters, in which both the motion and force control are covered.

2 Motion Representation

2.1 Spatial Motion Representation

Six independent parameters are sufficient to fully describe the spatial location of a rigid body.
Consider a rigid body in a spatial motion as represented in Figure 1. Let us define:

- A **fixed reference coordinate system** (x, y, z) denoted by frame $\{\boldsymbol{A}\}$ whose origin is located at point O_A
- A **moving coordinate system** (u, v, z) denoted by frame $\{\boldsymbol{B}\}$ attached to the rigid body at point O_B

The absolute position of point P of the rigid body can be constructed from the relative position of that point with respect to the moving frame $\{\boldsymbol{B}\}$, and the **position and orientation** of the moving frame $\{\boldsymbol{B}\}$ with respect to the fixed frame $\{\boldsymbol{A}\}$.

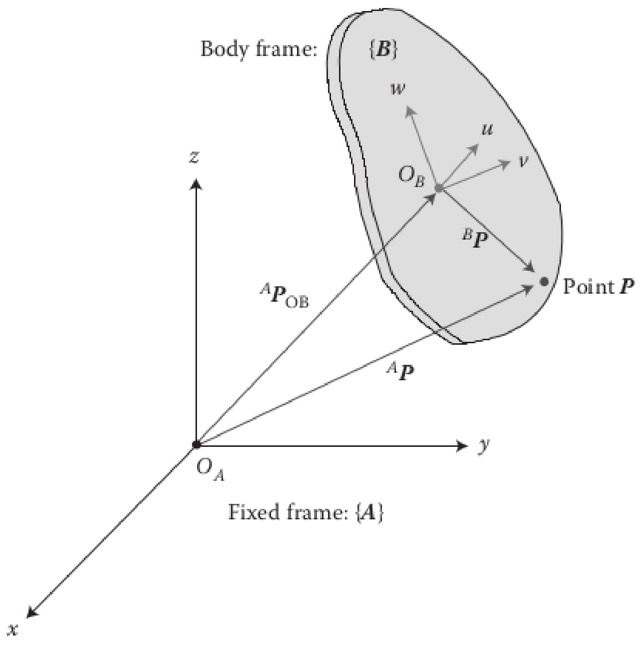

Figure 1 – *Representation of a rigid body spatial motion*

a Position of a point

The position of a point P with respect to a frame $\{\boldsymbol{A}\}$ can be described by a 3×1 position vector. The name of the frame is usually added as a leading superscript: ${}^A\boldsymbol{P}$ which reads as vector $\boldsymbol{P}$ in frame $\{\boldsymbol{A}\}$.

$$\boxed{{}^A\boldsymbol{P} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix}} \tag{1}$$

b Orientation of a Rigid Body

The orientation of the whole rigid body is the same for all its points (by definition). Hence, representation of the orientation of a rigid body can be viewed as that for the orientation of a moving frame attached to the rigid body. It can be **represented in several different ways**: the rotation matrix, the screw axis representation and Euler angles are common descriptions.

Rotation Matrix We consider a rigid body that has been exposed to a pure rotation. Its orientation has changed from a state represented by frame $\{\boldsymbol{A}\}$ to its current orientation represented by frame $\{\boldsymbol{B}\}$ (Figure 2).
A 3×3 rotation matrix ${}^A\boldsymbol{R}_B$ is defined by

$$\boxed{{}^A\boldsymbol{R}_B = \left[{}^A\hat{\boldsymbol{x}}_B | {}^A\hat{\boldsymbol{y}}_B | {}^A\hat{\boldsymbol{z}}_B\right] = \begin{bmatrix} u_x & v_x & z_x \\ u_y & v_y & z_y \\ u_z & v_z & z_z \end{bmatrix}} \tag{2}$$

in which ${}^A\hat{\boldsymbol{x}}_B, {}^A\hat{\boldsymbol{y}}_B$ and ${}^A\hat{\boldsymbol{z}}_B$ are the Cartesian unit vectors of frame $\{\boldsymbol{B}\}$ represented in frame $\{\boldsymbol{A}\}$.

$$\begin{aligned} {}^A\hat{\boldsymbol{x}}_B &= {}^A\hat{u} = u_x\hat{\imath} + u_y\hat{\jmath} + u_z\hat{k} \\ {}^A\hat{\boldsymbol{y}}_B &= {}^A\hat{v} = v_x\hat{\imath} + v_y\hat{\jmath} + v_z\hat{k} \\ {}^A\hat{\boldsymbol{z}}_B &= {}^A\hat{w} = w_x\hat{\imath} + w_y\hat{\jmath} + w_z\hat{k} \end{aligned}$$

The nine elements of the rotation matrix can be simply represented as the projections of the Cartesian unit vectors of frame $\{\boldsymbol{B}\}$ on the unit vectors of frame $\{\boldsymbol{A}\}$.

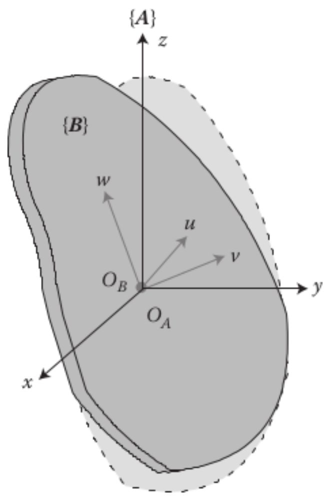

Figure 2 – *Pure rotation of a rigid body*

The rotation matrix has a number of properties linking each of its nine elements:

- **Orthonormality**: the rotation matrix is an orthonormal matrix
- **Transposition**: ${}^B\boldsymbol{R}_A = {}^A\boldsymbol{R}_B^T$
- **Inverse**: ${}^B\boldsymbol{R}_A = {}^A\boldsymbol{R}_B^{-1} = {}^A\boldsymbol{R}_B^T$
- **Pure Rotation Mapping**: Suppose that the point of a rigid body with respect to the moving frame $\{\boldsymbol{B}\}$ is given and denoted by ${}^B\boldsymbol{P}$ and we wish to express the position of this point with respect to the fixed frame $\{\boldsymbol{A}\}$. Consider that the rigid body has been exposed to a pure rotation ($\{\boldsymbol{A}\}$ and $\{\boldsymbol{B}\}$ are coincident at their origins). Then

$$\boxed{{}^A\boldsymbol{P} = {}^A\boldsymbol{R}_B {}^B\boldsymbol{P}}$$

- **Determinant**: $\det\left({}^A\boldsymbol{R}_B\right) = 1$
- **Eigenvalues**: The eigenvalues of a rotation matrix ${}^A\boldsymbol{R}_B$ are equal to 1, $e^{i\theta}$ and $e^{-i\theta}$ where θ is calculated from $\theta = \cos^{-1}\frac{\operatorname{tr}({}^A\boldsymbol{R}_B)-1}{2}$.

Screw Axis Representation As seen previously, there exist an **invariant angle** θ corresponding to the rotation matrix. This angle is an equivalent angle of rotation. The rotation is a spatial change of orientation about an axis which is called the **screw axis**. It can be shown that this screw axis is also an invariant of the rotation matrix, it is the eigenvector corresponding to the eigenvalue $\lambda = 1$.

The term screw axis for this axis of rotation has the benefit that a general motion of a rigid body, which is composed as a pure translation and a pure rotation, can be further represented by the same axis of rotation.

The screw axis representation has the benefit of **using only four parameters** to describe a pure rotation. These parameters are the angle of rotation θ and the axis of rotation which is a unit vector ${}^A\hat{\boldsymbol{s}} = [s_x, s_y, s_z]^T$.

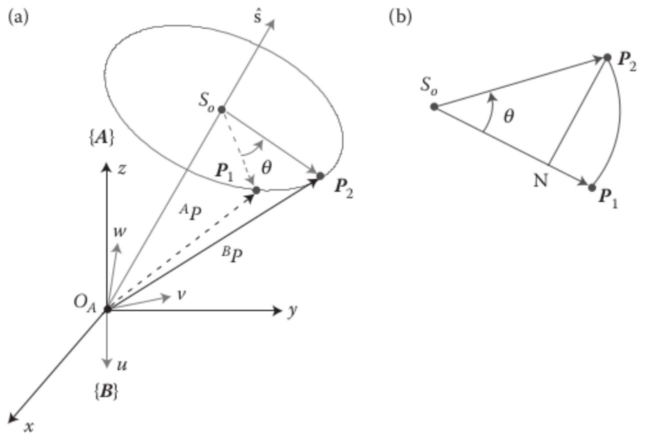

Figure 3 – *Pure rotation about a screw axis*

The Rodrigue's rotation formula for spatial rotation of a rigid body gives us the new position $\boldsymbol{P}_2$ of point $\boldsymbol{P}_1$ after a rotation represented by the screw axis $\hat{\boldsymbol{s}}$ and the angle θ:

$$\boxed{\boldsymbol{P}_2 = \boldsymbol{P}_1 \cos\theta + (\hat{\boldsymbol{s}} \times \boldsymbol{P}_1)\sin\theta + (\boldsymbol{P}_1 \cdot \hat{\boldsymbol{s}})\hat{\boldsymbol{s}}} \quad (3)$$

Euler Angles Since rotation in space is a motion with three-degrees-of-freedom, a set of three independent parameters is sufficient to represent the orientation.

In an Euler angle representation, three **successive** rotations about the coordinate system of either **fixed** or **moving** frame are used to describe the orientation of the rigid body.

One type of Euler angle corresponds to rotations considered with respect to the fixed frame. The representation is called pitch-roll-yaw, or fixed X-Y-Z Euler angles.

Three other types of Euler angles are consider with respect to a moving frame: they are denoted $w-v-u$, $w-v-w$ and $w-u-w$ Euler angles.

c Pitch-Roll-Yaw Euler Angles

The pitch, roll and yaw angles are defined for a moving object in space as the rotations along the lateral, longitudinal and vertical axes attached to the moving object.

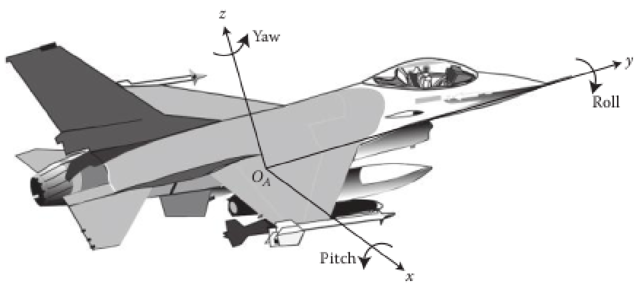

Figure 4 – *Definition of pitch, roll and yaw angles on an air plain*

Since all three rotations take place about the axes of a **fixed coordinate frame**, the resulting rotation matrix is obtained by multiplying the three basic rotation matrices as follows:

$$\boxed{\boldsymbol{R}_{PRY}(\alpha,\beta,\gamma) = \boldsymbol{R}_z(\gamma)\boldsymbol{R}_y(\beta)\boldsymbol{R}_x(\alpha)}$$

To go from rotation matrix to Pitch-Roll-Yaw angles, the following set of equations can be used:

$$\alpha = \operatorname{atan2}\left(\frac{r_{32}}{\cos\beta}, \frac{r_{33}}{\cos\beta}\right)$$

$$\beta = \operatorname{atan2}\left(-r_{31}, \pm\sqrt{r_{11}^2 + r_{21}^2}\right)$$

$$\gamma = \operatorname{atan2}\left(\frac{r_{21}}{\cos\beta}, \frac{r_{11}}{\cos\beta}\right)$$

d u-v-w Euler Angles

Another way to describe the orientation of a moving object is to consider three successive rotations about the **coordinate axes of the moving frame**. Since the rotations do not occur about fixed axes, pre-multiplications of the individual rotation matrices fails to give the correct solution. It can be shown that the resulting matrix can be derived by post-multiplication of the individual rotation matrices as follows:

$$^{A}\boldsymbol{R}_B(\alpha,\beta,\gamma) = \boldsymbol{R}_u(\alpha)\boldsymbol{R}_v(\beta)\boldsymbol{R}_w(\gamma)$$

The inverse solution for the u-v-w Euler angles is the following (for $\cos\beta \neq 0$):

$$\alpha = \operatorname{atan2}\left(-\frac{r_{23}}{\cos\beta}, \frac{r_{33}}{\cos\beta}\right)$$
$$\beta = \operatorname{atan2}\left(r_{13}, \pm\sqrt{r_{11}^2 + r_{12}^2}\right)$$
$$\gamma = \operatorname{atan2}\left(-\frac{r_{12}}{\cos\beta}, \frac{r_{11}}{\cos\beta}\right)$$

e w-v-w Euler Angles

Similarly:

$$\boldsymbol{R}_{wvw}(\alpha,\beta,\gamma) = \boldsymbol{R}_w(\alpha)\boldsymbol{R}_v(\beta)\boldsymbol{R}_w(\gamma)$$

And for $\sin\beta \neq 0$:

$$\alpha = \operatorname{atan2}\left(\frac{r_{23}}{\sin\beta}, \frac{r_{13}}{\sin\beta}\right)$$
$$\beta = \operatorname{atan2}\left(\pm\sqrt{r_{31}^2 + r_{32}^2}, r_{33}\right)$$
$$\gamma = \operatorname{atan2}\left(\frac{r_{32}}{\sin\beta}, -\frac{r_{31}}{\sin\beta}\right)$$

f w-u-w Euler Angles

Here, the second rotation is about the u axis:

$$\boldsymbol{R}_{wuw}(\alpha,\beta,\gamma) = \boldsymbol{R}_w(\alpha)\boldsymbol{R}_u(\beta)\boldsymbol{R}_w(\gamma)$$

And for $\sin\beta \neq 0$:

$$\alpha = \operatorname{atan2}\left(\frac{r_{13}}{\sin\beta}, -\frac{r_{23}}{\sin\beta}\right)$$
$$\beta = \operatorname{atan2}\left(\pm\sqrt{r_{31}^2 + r_{32}^2}, r_{33}\right)$$
$$\gamma = \operatorname{atan2}\left(\frac{r_{31}}{\sin\beta}, \frac{r_{32}}{\sin\beta}\right)$$

g Notes about Euler Angles

If the Euler angle is given, a **unique rotation matrix** is determined for the orientation of the rigid body. However, the inverse map is not one-to-one, and at least two Euler angle sets can be found for each orientation. If the Euler angle is chosen for the representation of the orientation, extra care should be taken. From the continuity of the motion, a suitable solution may be chosen, such that no abrupt changes are seen in the variation of the Euler angles in a typical maneuver.
The use of rotation matrix to represent the orientation of a rigid body is then generally preferred although there are nine parameters for that description.

2.2 Motion of a Rigid Body

Since the relative positions of a rigid body with respect to a moving frame $\{\boldsymbol{B}\}$ attached to it is fixed for all time, it is sufficient to know the **position of the origin of the frame** O_B and the **orientation of the frame** $\{\boldsymbol{B}\}$ with respect to the fixed frame $\{\boldsymbol{A}\}$, to represent the position of any point P in the space.

Representation of the position of O_B is uniquely given by the position vector, while orientation of the rigid body is represented in different forms. However, for all possible orientation representation, a rotation matrix $^{A}\boldsymbol{R}_B$ can be derived.

Pose of a rigid body

Therefore, the location or **pose** of a rigid body, can be **fully determined** by:

1. The **position vector** of point O_B with respect to frame $\{\boldsymbol{A}\}$ which is denoted $^{A}\boldsymbol{P}_{O_B}$
2. The **orientation of the rigid body**, or the moving frame $\{\boldsymbol{B}\}$ attached to it with respect to the fixed frame $\{\boldsymbol{A}\}$, that is represented by $^{A}\boldsymbol{R}_B$.

The position of any point P of the rigid body with respect to the fixed frame $\{\boldsymbol{A}\}$, which is denoted $^{A}\boldsymbol{P}$ may be determined thanks to the Chasles' theorem.

Chasles' theorem

If the pose of a rigid body $\{^{A}\boldsymbol{R}_B, {}^{A}\boldsymbol{P}_{O_B}\}$ is given, then the position of any point P of this rigid body with respect to $\{\boldsymbol{A}\}$ is given by:

$$^{A}\boldsymbol{P} = {}^{A}\boldsymbol{R}_B {}^{B}\boldsymbol{P} + {}^{A}\boldsymbol{P}_{O_B} \tag{4}$$

2.3 Homogeneous Transformations

To describe general transformations, we introduce the 4×1 **homogeneous coordinates**, and Eq. (4) is generalized to

$$\boxed{^{A}\boldsymbol{P} = {}^{A}\boldsymbol{T}_B {}^{B}\boldsymbol{P}} \tag{5}$$

in which $^{A}\boldsymbol{T}_B$ is a 4×4 **homogeneous transformation matrix**.

a Homogeneous Coordinates

There are two basic classes of vector quantities, the generalization to homogeneous coordinates of which are different.

The first type is called **line vector**. Line vectors refer to a vector of which its value depends on the line of action, or the position of where it is applied. Examples are the position vector, linear velocity, force vector.

On the contrary, there exist quantities likes orientation that **hold for the whole rigid body** and do not correspond to a point. They can be positioned freely throughout the whole rigid body, without any change in their quantity. These types of vectors are called **free vectors**.

For line vectors, both orientation and translation of the moving frame contribute to their value. Homogeneous coordinate of such vectors is generated by appending 1 to the three components of that vector:

$$\boxed{\boldsymbol{V} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ 1 \end{bmatrix}} \tag{6}$$

For free vectors, only the orientation of the moving frame contributes to their value. The homogeneous coordinate is then

$$\boxed{\boldsymbol{\omega} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \\ 0 \end{bmatrix}} \tag{7}$$

b Homogeneous Transformation Matrix

Homogeneous Transformation Matrix

The **homogeneous transformation matrix** is a 4×4 matrix, defined for the purpose of transformation mapping of a vector in a homogeneous coordinate from one frame to another in a compact form. The matrix is composed of the rotation matrix ${}^A\boldsymbol{R}_B$ representing the orientation and the position vector ${}^A\boldsymbol{P}_{O_B}$ representing the translation. It is partitioned as follows:

$${}^A\boldsymbol{T}_B = \left[\begin{array}{ccc|c} & {}^A\boldsymbol{R}_B & & {}^A\boldsymbol{P}_{O_B} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \tag{8}$$

The homogeneous transformation matrix ${}^A\boldsymbol{T}_B$ is a 4×4 matrix operator mapping **vector valued** quantities represented by 4×1 homogeneous coordinates.:

$$\left[\begin{array}{c} {}^A\boldsymbol{P} \\ \hline 1 \end{array}\right] = \left[\begin{array}{ccc|c} & {}^A\boldsymbol{R}_B & & {}^A\boldsymbol{P}_{O_B} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \left[\begin{array}{c} {}^B\boldsymbol{P} \\ \hline 1 \end{array}\right]$$

$${}^A\boldsymbol{P} = {}^A\boldsymbol{R}_B{}^B\boldsymbol{P} + {}^A\boldsymbol{P}_{O_B}$$

Using homogeneous coordinate for a **free vector** like angular velocity of a rigid body:

$$\left[\begin{array}{c} {}^A\boldsymbol{\omega} \\ \hline 0 \end{array}\right] = \left[\begin{array}{ccc|c} & {}^A\boldsymbol{R}_B & & {}^A\boldsymbol{P}_{O_B} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \left[\begin{array}{c} {}^B\boldsymbol{\omega} \\ \hline 0 \end{array}\right]$$

$${}^A\boldsymbol{P} = {}^A\boldsymbol{R}_B{}^B\boldsymbol{P}$$

c Screw Displacement

The most general rigid body displacement can be produced by a **translation along a line followed by a rotation about the same line**. The line is called the **screw axis**.
There exist transformations to from screw displacement notation to the transformation matrix.

d Transformation Arithmetics

Consecutive transformations Let us consider the motion of a rigid body described at three locations (Figure 5). Frame $\{\boldsymbol{A}\}$ represents the initial location, frame $\{\boldsymbol{B}\}$ is an intermediate location, and frame $\{\boldsymbol{C}\}$ represents the rigid body at its final location.

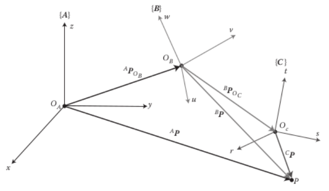

Figure 5 – *Motion of a rigid body represented at three locations by frame $\{\boldsymbol{A}\}$, $\{\boldsymbol{B}\}$ and $\{\boldsymbol{C}\}$*

Furthermore, suppose the position vector of a point P of the rigid body is given in the final location, that is ${}^C\boldsymbol{P}$ is given, and the position of this point is to be found in the fixed frame $\{\boldsymbol{A}\}$, that is ${}^A\boldsymbol{P}$. Since the locations of the rigid body is known relative to each other, ${}^C\boldsymbol{P}$ can be transformed to ${}^B\boldsymbol{P}$ using ${}^B\boldsymbol{T}_C$:

$${}^B\boldsymbol{P} = {}^B\boldsymbol{T}_C{}^C\boldsymbol{P}$$

Now, ${}^B\boldsymbol{P}$ can be transformed into ${}^A\boldsymbol{P}$:

$${}^A\boldsymbol{P} = {}^A\boldsymbol{T}_B{}^B\boldsymbol{P}$$

And we have:

$${}^A\boldsymbol{P} = {}^A\boldsymbol{T}_B{}^B\boldsymbol{T}_C{}^C\boldsymbol{P}$$

From which, the consecutive transformation can be defined as follows:

$$\boxed{{}^{A}\boldsymbol{T}_{C} = {}^{A}\boldsymbol{T}_{B}\,{}^{B}\boldsymbol{T}_{C}} \tag{9}$$

Inverse transformation Direct inversion of the 4×4 homogeneous transfer matrix ${}^{A}\boldsymbol{T}_{B}$ to obtain ${}^{B}\boldsymbol{T}_{A}$ might be computationally intensive. It is much easier to use the specific structure of the transfer matrix for inversion. To obtain ${}^{B}\boldsymbol{T}_{A}$, it is necessary to compute ${}^{B}\boldsymbol{R}_{A}$ and ${}^{B}\boldsymbol{P}_{O_A}$ from the known ${}^{A}\boldsymbol{R}_{B}$ and ${}^{A}\boldsymbol{P}_{O_B}$, then

$${}^{B}\boldsymbol{T}_{A} = \left[\begin{array}{ccc|c} & {}^{B}\boldsymbol{R}_{A} & & {}^{B}\boldsymbol{P}_{O_A} \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

Moreover

$$\begin{aligned} {}^{B}\boldsymbol{R}_{A} &= {}^{A}\boldsymbol{R}_{B}^{T} \\ {}^{B}\boldsymbol{P}_{O_A} &= {}^{B}\boldsymbol{R}_{A}\,{}^{A}\boldsymbol{P}_{O_A} = -{}^{B}\boldsymbol{R}_{A}\,{}^{A}\boldsymbol{P}_{O_B} \\ &= -{}^{A}\boldsymbol{R}_{B}^{T}\,{}^{A}\boldsymbol{P}_{O_B} \end{aligned}$$

Hence, the **inverse of the transformation matrix** can be obtain by

$${}^{B}\boldsymbol{T}_{A} = {}^{A}\boldsymbol{T}_{B}^{-1} = \left[\begin{array}{ccc|c} & {}^{A}\boldsymbol{R}_{B}^{T} & & -{}^{A}\boldsymbol{R}_{B}^{T}\,{}^{A}\boldsymbol{P}_{O_B} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \tag{10}$$

3 Kinematics

3.1 Introduction

Kinematic Analysis - Definition

Kinematic analysis refers to the study of the geometry of motion of a robot, without considering the forces an torques that cause the motion. In this analysis, the relation between the geometrical parameters of the manipulator with the final motion of the moving platform is derived and analyzed.

A **parallel robot** is a mechanism with a number of **closed kinematic chains**, and its moving platform is linked to the base by several independent kinematic chains. Parallel robots for which the number of kinematic chains is equal to the number of degrees-of-freedom of the moving platform are called **fully parallel robots**.

If in addition to this condition, if the type and number of joints at each limb, and the number and location of the actuated joints are identical in all the limbs, such a parallel robot is called **symmetric**.

There are three main cases for fully parallel manipulators. Planar robots with two translation and one rotational degree-of-freedom in the plane. Spatial orientation manipulators with three rotational degrees-of-freedom in space. And a general spatial robot with three translational and three rotational degrees-of-freedom in space.

It is known that unlike serial manipulators, **inverse kinematic analysis of parallel robots is usually simple and straightforward**. In most cases, limb variable may be computed independently using the given pose of the moving platform, and the solution in most cases even for redundant manipulators is uniquely determined. However, **forward kinematics of parallel manipulators is generally very complicated**, and its solution usually involves systems of nonlinear equations, which are highly coupled and in general have no closed form and unique solution.

3.2 Loop Closure Method

A typical parallel manipulator consists of two main bodies. Body A is arbitrary designated as fixed and is called the **base**, while body B is designated to be movable and is called the **moving platform**.
These two bodies are coupled via n **limbs**, each attached to points A_i and B_i and called fixed and moving attachment points of the limb i.

At the **displacement** level, the **forward kinematic** problem permits the determination of the actual location or pose of the moving platform relative to the base from a set of joint-position readouts.
At the **velocity** level, the **forward kinematic** problem refers to the determination of the translational and angular velocities of the moving platform relative to the base, from a set of joint-velocity readouts and for a known configuration.

To describe the motion of the moving platform relative to the base, frame $\{\boldsymbol{A}\}$ is attached to body A and frame $\{\boldsymbol{B}\}$ to body B. The pose of the moving platform relative to the base is thus defined by:

- A position vector $\boldsymbol{p}$ which denotes the position vector of the origin of $\{\boldsymbol{B}\}$ with respect to frame $\{\boldsymbol{A}\}$
- A 3×3 rotation matrix R which denotes the rotation of $\{\boldsymbol{B}\}$ with respect to $\{\boldsymbol{A}\}$

Each limb of a parallel manipulator defines a kinematic loop passing through the origins of frames $\{\boldsymbol{A}\}$ and $\{\boldsymbol{B}\}$, and through the two limb attachment points A_i and B_i.
At the displacement level, the **closure of each kinematic loop** can be express in the vector form as

$$\vec{AB} = \vec{AA_i} + \vec{A_iB_i} - \vec{BB_i} \quad \text{for } i = 1, 2, \dots, n$$

in which $\vec{AA_i}$ and $\vec{BB_i}$ can be easily obtained from the geometry of the attachment points in the base and in the moving platform.
Let us defined $\boldsymbol{a}_i = \vec{AA_i}$ in the fixed frame $\{\boldsymbol{A}\}$, and $\boldsymbol{b}_i = \vec{BB_i}$ in the moving frame $\{\boldsymbol{B}\}$. Furthermore, $\boldsymbol{q}_i = \vec{A_iB_i}$ is defined as the **limb variable**, which indicated the geometry of the limb.

Loop Clusure

The **loop closure** can be written as the unknown pose variables $\boldsymbol{p}$ and $\boldsymbol{R}$, the position vectors describing the known geometry of the base and of the moving platform, $\boldsymbol{a}_i$ and $\boldsymbol{b}_i$, and the limb vector $\boldsymbol{q}_i$

$$\boldsymbol{p} = \boldsymbol{a}_i + \boldsymbol{q}_i - \boldsymbol{R}\boldsymbol{b}_i \quad \text{for } i = 1, 2, \dots, n \tag{11}$$

For an **inverse kinematic problem**, it is assumed that the moving platform position $\boldsymbol{p}$ and orientation $\boldsymbol{R}$ are given and the problem is to solve the active limb variables. This analysis is usually straightforward and results in unique solution for the limb variables.
However, the inverse solution is not straightforward, and usually numerical methods are used for forward kinematic solution.

3.3 Kinematic Analysis of a Stewart-Gough Platform

a Mechanism Description

One frame $\{\boldsymbol{A}\}$ is attached to the fixed base and frame $\{\boldsymbol{B}\}$ is attached to the moving platform at points O_A and O_B respectively.
The number of actuators is equal to the degrees-of-freedom of the manipulator and hence the manipulator is **fully parallel**.

Force Orientation

Since all the limbs are connected to the moving platform and to the base by spherical joints, **no twisting torque** can be transmitted and the force acting on each limb is directed along the longitudinal axis of the limb.

b Geometry of the Manipulator

Geometry of the Manipilator

The position of the attachment points on the fixed base are denoted by the vectors $\boldsymbol{a}_i$ and the position of moving attachment points are denoted by the vectors $\boldsymbol{b}_i$. The geometry of each limb is described by its length l_i and its direction is denoted by a unit vector $\hat{\boldsymbol{s}}_i$.

The position of the point O_B of the moving platform is described by the **position vector** ${}^A\boldsymbol{P} = [p_x\ p_y\ p_z]^T$ and orientation of the moving platform is described by the **rotation matrix** ${}^A\boldsymbol{R}_B$ which can by represented by the components of the unit vectors $\hat{\boldsymbol{u}}$, $\hat{\boldsymbol{v}}$, $\hat{\boldsymbol{z}}$ as follows:

$$ {}^A\boldsymbol{R}_B = \begin{bmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{bmatrix} \tag{12}$$

The geometry of the manipulator is shown Figure 6.

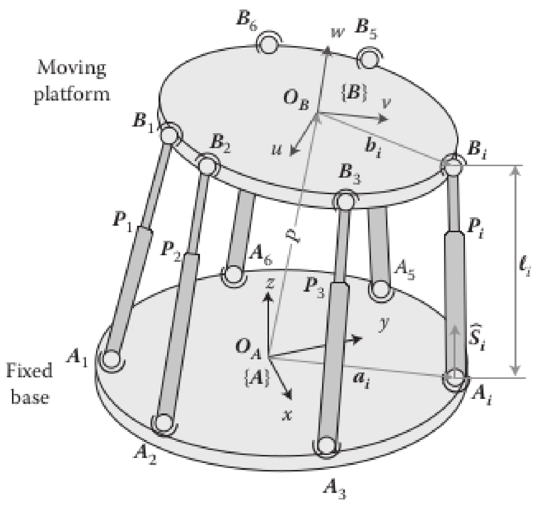

Figure 6 – *Geometry of a Stewart-Gough platform*

c Inverse Kinematics

Inverse Kinematic Analysis

For **inverse kinematic analysis**, it is assumed that the position ${}^A\boldsymbol{P}$ and orientation of the moving platform ${}^A\boldsymbol{R}_B$ are given and the problem is to obtain the joint variables $\boldsymbol{L} = [l_1, l_2, l_3, l_4, l_5, l_6]^T$.

From the geometry of the manipulator, one can write:

$$ {}^A\boldsymbol{a}_i + l_i {}^A\hat{\boldsymbol{s}}_i = {}^A\boldsymbol{P} + {}^A\boldsymbol{b}_i \tag{13}$$

Then, we can find l_i given ${}^A\boldsymbol{P}$ and ${}^A\boldsymbol{R}_B$:

$$ \begin{aligned} l_i = \Big[& {}^A\boldsymbol{P}^T {}^A\boldsymbol{P} + {}^B\boldsymbol{b}_i^T {}^B\boldsymbol{b}_i + {}^A\boldsymbol{a}_i^T {}^A\boldsymbol{a}_i - 2 {}^A\boldsymbol{P}^T {}^A\boldsymbol{a}_i + \dots \\ & 2 {}^A\boldsymbol{P}^T \left[{}^A\boldsymbol{R}_B {}^B\boldsymbol{b}_i \right] - 2 \left[{}^A\boldsymbol{R}_B {}^B\boldsymbol{b}_i \right]^T {}^A\boldsymbol{a}_i \Big]^{1/2} \end{aligned} \tag{14}$$

If the position and orientation of the platform lie in the feasible workspace, the solution is unique. Otherwise, the solution gives complex numbers.

d Forward Kinematics

Forward Kinematic Analysis

In **forward kinematic analysis**, it is assumed that the vector of limb lengths $\boldsymbol{L}$ is given and the problem is to find the position ${}^A\boldsymbol{P}$ and the orientation ${}^A\boldsymbol{R}_B$.

The size of the problem depends of the representation used for orientation (rotation matrix, Euler angles, ...).
The forward kinematic problem is then to solve many **highly nonlinear equations** that are extremely difficult to solve.
The complexity of the problem depends widely on the manipulator architecture and geometry.

4 Jacobian: Velocities and Static Forces

4.1 Introduction

Usefullness of Jacobian matrix

The Jacobian matrix not only reveals the **relation between the joint variable velocities of a parallel manipulator to the moving platform linear and angular velocities**, it also constructs the transformation needed to find the **actuator forces from the forces and moments acting on the moving platform**.

For specific configurations, **local degeneracy** can occur and leads to:

1. An instantaneous change in the degrees-of-freedom of the system and hence a **loss of controllability**
2. An important **degradation of the natural stiffness** that may lead to very high joint forces or torques

Therefore, it is very important to **identify singular configurations** at the design stage to improve the performance.

4.2 Angular and Linear Velocities

To determine the absolute linear velocity of a point, the derivative must be calculated relative to a **fixed** frame. Differentiation of a position vector with respect to a moving frame results in a relative velocity. Therefore, it is necessary to define the arithmetics to transform the relative velocities to the absolute ones.

a Angular Velocity of a Rigid Body

Angular velocity is an attribute of a rigid body that describes the rotational motion of the frame $\{\boldsymbol{B}\}$ that is attached to the rigid body.

Angular Velocity Vector

The **angular velocity vector $\boldsymbol{\Omega}$** describes the instantaneous rotation of frame $\{\boldsymbol{B}\}$ with respect to the fixed frame $\{\boldsymbol{A}\}$. The direction of $\boldsymbol{\Omega}$ indicates the instantaneous axis of rotation and its magnitude indicates the speed of rotation.

The angular velocity vector is related to the screw formalism by equation (15).

$$\boxed{\boldsymbol{\Omega} \triangleq \dot{\theta}\hat{\boldsymbol{s}}} \tag{15}$$

The angular velocity can be expressed in any frame. For example ${}^A\boldsymbol{\Omega}$ denotes the angular velocity of the rigid body expressed in the frame $\{\boldsymbol{A}\}$ and we have:

$$\begin{aligned} {}^A\boldsymbol{\Omega} &= \Omega_x\hat{\boldsymbol{x}} + \Omega_y\hat{\boldsymbol{y}} + \Omega_z\hat{\boldsymbol{z}} \\ &= \dot{\theta}\left(s_x\hat{\boldsymbol{x}} + s_y\hat{\boldsymbol{y}} + s_z\hat{\boldsymbol{z}}\right) \end{aligned} \tag{16}$$

in which Ω_x, Ω_y and Ω_z are the three components of angular velocity of a rigid body expressed in frame $\{\boldsymbol{A}\}$.

b Linear Velocity of a Point

Linear velocity of a point P can be easily determined by the time derivative of the position vector p of that point with respect to a fixed frame:

$$v_p = \dot{p} = \left(\frac{\mathrm{d}p}{\mathrm{d}t}\right)_{\text{fix}} \tag{17}$$

If the variation of the position vector is determined with respect to a moving frame, we obtain the relative velocity:

$$v_{\text{rel}} = \left(\frac{\partial p}{\partial t}\right)_{\text{mov}} \tag{18}$$

In classical mechanics, it is shown that the relation between absolute derivative of any vector to its relative derivative is given by:

$$\left(\frac{\mathrm{d}(\cdot)}{\mathrm{d}t}\right)_{\text{fix}} = \left(\frac{\partial(\cdot)}{\partial t}\right)_{\text{mov}} + \boldsymbol{\Omega}\times(\cdot) \tag{19}$$

in which $\boldsymbol{\Omega}$ denotes the angular velocity of the moving frame with respect to the fixed frame.
The term $\boldsymbol{\Omega}\times(\cdot)$ can be written in matrix form:

$$\boxed{\left(\frac{\mathrm{d}(\cdot)}{\mathrm{d}t}\right)_{\text{fix}} = \left(\frac{\partial(\cdot)}{\partial t}\right)_{\text{mov}} + \boldsymbol{\Omega}^\times(\cdot)} \tag{20}$$

The matrix $\boldsymbol{\Omega}^\times$ denotes a **skew-symmetric matrix** defined by:

$$\boxed{\boldsymbol{\Omega}^\times = \begin{bmatrix} 0 & -\Omega_z & \Omega_y \\ \Omega_z & 0 & -\Omega_x \\ -\Omega_y & \Omega_x & 0 \end{bmatrix}} \tag{21}$$

Now consider the general motion of a rigid body shown in Figure 7, in which a moving frame $\{\boldsymbol{B}\}$ is attached to the rigid body and **the problem is to find the absolute velocity** of point P with respect to a fixed frame $\{\boldsymbol{A}\}$.
The rigid body perform a general motion which is a combination of a translation, denoted by the vector ${}^A\boldsymbol{P}_{O_B}$, and an instantaneous rotation. To determine the velocity of point P, we start with the relation between absolute and relative position vectors:

$${}^A\boldsymbol{P} = {}^A\boldsymbol{P}_{O_B} + {}^A\boldsymbol{R}_B{}^B\boldsymbol{P}$$

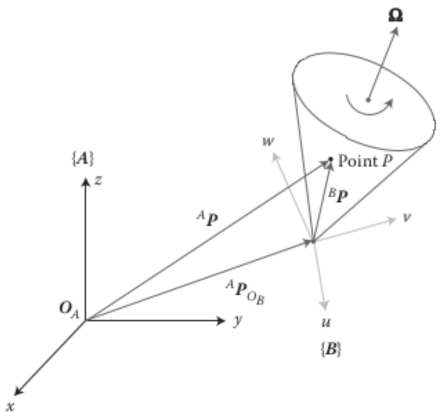

Figure 7 – *Instantaneous velocity of a point P with respect to a moving frame $\{\boldsymbol{B}\}$*

To derive the velocity of point P, we differentiate with respect to time:

$$ {}^A\boldsymbol{v}_P = {}^A\boldsymbol{v}_{O_B} + {}^A\dot{\boldsymbol{R}}_B {}^B\boldsymbol{P} + {}^A\boldsymbol{R}_B \underbrace{{}^B\boldsymbol{v}_P}_{=0} $$

The time derivative of the rotation matrix ${}^A\dot{\boldsymbol{R}}_B$ is:

$$ \boxed{{}^A\dot{\boldsymbol{R}}_B = {}^A\boldsymbol{\Omega}^\times \, {}^A\boldsymbol{R}_B} \tag{22} $$

And we finally obtain equation (23).

Absolute Linear Velocity

$$ {}^A\boldsymbol{v}_P = {}^A\boldsymbol{v}_{O_B} + {}^A\boldsymbol{\Omega}^\times \, {}^A\boldsymbol{R}_B {}^B\boldsymbol{P} \tag{23} $$

c Screw Coordinates

Finite rotation of a rigid body can be expressed as a rotation θ about a screw axis $\hat{\boldsymbol{s}}$. Furthermore, it is shown that the angular velocity of a rigid body is also defined as the rate of instantaneous rotation angle $\dot{\theta}$ about the same screw axis $\hat{\boldsymbol{s}}$.

Chasles' theorem

The most general rigid-body displacement can be produced by a translation along a line followed by a rotation about the same line. Since this displacement is reminiscent of the displacement of a screw, it is called a **screw displacement**, and the line of axis is called the **screw axis**.

4.3 Jacobian Matrices of a Parallel Manipulator

Let $\boldsymbol{q} = [q_1, q_2, \dots, q_m]^T$ denote the vector of actuated joint coordinates (linear displacement of an actuator prismatic joint or angular rotation of an actuated revolute joint) and $\boldsymbol{\mathcal{X}} = [x_1, x_2, \dots, x_n]^T$ denote the vector of moving platform motion variables (position or orientation).
m denotes the number of actuated joints in the manipulator, n denotes the number of degrees-of-freedom of the manipulator.
Generally $m \geq n$, in which for a fully parallel manipulator $m = n$ and for redundant manipulator $m > n$.

$\boldsymbol{q}$ and $\boldsymbol{\mathcal{X}}$ are related through a system of **nonlinear algebraic equations** representing the **kinematic constraints imposed by the limbs**, which can be generally written as

$$ f(\boldsymbol{q}, \boldsymbol{\mathcal{X}}) = 0 \tag{24} $$

We can differentiate this equation with respect to time and obtain:

$$ \boxed{\boldsymbol{J}_x \dot{\boldsymbol{\mathcal{X}}} = \boldsymbol{J}_q \dot{\boldsymbol{q}}} \tag{25} $$

where

$$ \boldsymbol{J}_x = \frac{\partial f}{\partial \boldsymbol{\mathcal{X}}} \quad \text{and} \quad \boldsymbol{J}_q = -\frac{\partial f}{\partial \boldsymbol{q}} \tag{26} $$

General Jacobian matrix J

The **general Jacobian matrix** is defined as:

$$ \dot{\boldsymbol{q}} = \boldsymbol{J} \dot{\boldsymbol{\mathcal{X}}} \tag{27} $$

From equation (26), we have:

$$ \boldsymbol{J} = \boldsymbol{J}_q^{-1} \boldsymbol{J}_x \tag{28} $$

4.4 Velocity Loop Closure

The **velocity loop closures** are used for **obtaining the Jacobian matrices** in a straightforward manner. Velocity loop closures are derived by direct differentiation of kinematic loop closures.

Kinematic loop closures are:

$$ \boldsymbol{p} = \boldsymbol{a}_i + \boldsymbol{d}_i - \boldsymbol{R}\boldsymbol{b}_i \quad \text{for } i = 1, \dots, m $$

with

- $\boldsymbol{p}$ the position vector of the moving platform w.r.t. frame $\{\boldsymbol{A}\}$
- $\boldsymbol{R}$ the rotation matrix of the moving platform
- $\boldsymbol{a}_i$ the position vector of the i'th limb of the fixed platform w.r.t. frame $\{\boldsymbol{A}\}$
- $\boldsymbol{b}_i$ the position vector of the i'th limb of the moving platform w.r.t. frame $\{\boldsymbol{B}\}$
- $\boldsymbol{d}_i$ the limb vector

By taking the time derivative, we obtain the following **Velocity Loop Closure**:

$$ \boxed{\dot{\boldsymbol{p}} = \dot{\boldsymbol{d}}_i - \boldsymbol{\omega} \times \boldsymbol{R}\boldsymbol{b}_i \quad \text{for } i = 1, \dots, m} \tag{29} $$

4.5 Singularity Analysis of Parallel Manipulators

The singularities occur when:

- $\boldsymbol{J}_q$ is rank deficient (Inverse kinematic singularity)
- $\boldsymbol{J}_x$ is rank deficient (Forward kinematic singularity)

a Inverse Kinematic Singularity

Inverse kinematic singularity happens when $\boldsymbol{J}_q$ ($m \times m$ matrix) is rank deficient ($\det \boldsymbol{J}_q = 0$).
The corresponding configurations are located at the boundary of the manipulator workspace or on the internal boundaries between sub-regions of the workspace.
In such cases, there exist nonzero vectors $\dot{\boldsymbol{q}}$ which correspond to a null Cartesian twist vector $\dot{\boldsymbol{\mathcal{X}}}$. In other words, infinitesimal motion of the moving platform along certain directions cannot be accomplished.
The manipulator looses one or more degrees-of-freedom.

b Forward Kinematic Singularity

Forward kinematic singularity happens when $\boldsymbol{J}_x$ ($m \times n$ matrix) is rank deficient ($\det\left(\boldsymbol{J}_x{}^T \boldsymbol{J}_x\right) = 0$).
If the manipulator is not redundantly actuated ($m = n$), then the Jacobian matrix $\boldsymbol{J}_x$ is square and the forward kinematic singularity happens when $\det \boldsymbol{J}_x = 0$.

The degeneracy occur inside the manipulator's Cartesian workspace and corresponds to the set of configurations for which two different branches of forward kinematic problem meet.

There exist nonzero cartesian twist vectors $\dot{\boldsymbol{\mathcal{X}}}$ that are mapped into a vanishing actuator velocity vector $\dot{\mathbf{\Pi}}$.
The corresponding configuration will be one in which an infinitesimal motion of the platform is possible even if the actuator are locked. **The manipulator gains one or several degrees-of-freedom and its stiffness vanishes in the corresponding direction(s)**.

4.6 Jacobian Analysis of the Stewart-Gough Platform

a Velocity Loop Closure

The input joint rate is denoted by $\dot{\boldsymbol{\mathcal{L}}} = [\dot{l}_1, \dot{l}_2, \dot{l}_3, \dot{l}_4, \dot{l}_5, \dot{l}_6]^T$, and the output twist vector is denoted by $\dot{\boldsymbol{\mathcal{X}}} = [{}^A\boldsymbol{v}_p, {}^A\boldsymbol{\omega}]^T$.

The jacobian matrix can be derived by formulating a velocity loop closure equation of each limb. The loop closure equations for each limb are:

$$ {}^A\boldsymbol{P} + {}^A\boldsymbol{R}_B {}^B\boldsymbol{b}_i = l_i {}^A\hat{\boldsymbol{s}}_i + {}^A\boldsymbol{a}_i \tag{30}$$

By differentiate this with respect to time:

$$ {}^A\boldsymbol{v}_p + {}^A\dot{\boldsymbol{R}}_B {}^B\boldsymbol{b}_i = \dot{l}_i {}^A\hat{\boldsymbol{s}}_i + l_i {}^A\dot{\hat{\boldsymbol{s}}}_i \tag{31}$$

Moreover, we have:

- ${}^A\dot{\boldsymbol{R}}_B {}^B\boldsymbol{b}_i = {}^A\boldsymbol{\omega} \times {}^A\boldsymbol{R}_B {}^B\boldsymbol{b}_i = {}^A\boldsymbol{\omega} \times {}^A\boldsymbol{b}_i$ in which ${}^A\boldsymbol{\omega}$ denotes the angular velocity of the moving platform expressed in the fixed frame $\{\boldsymbol{A}\}$.
- $l_i {}^A\dot{\hat{\boldsymbol{s}}}_i = l_i \left({}^A\boldsymbol{\omega}_i \times \hat{\boldsymbol{s}}_i\right)$ in which ${}^A\boldsymbol{\omega}_i$ is the angular velocity of limb i express in fixed frame $\{\boldsymbol{A}\}$.

Then, the velocity loop closure (31) simplifies to

$$ {}^A\boldsymbol{v}_p + {}^A\boldsymbol{\omega} \times {}^A\boldsymbol{b}_i = \dot{l}_i {}^A\hat{\boldsymbol{s}}_i + l_i ({}^A\boldsymbol{\omega}_i \times \hat{\boldsymbol{s}}_i)$$

By dot multiply both side of the equation by $\hat{\boldsymbol{s}}_i$:

$$ \hat{\boldsymbol{s}}_i {}^A\boldsymbol{v}_p + ({}^A\boldsymbol{b}_i \times \hat{\boldsymbol{s}}_i) {}^A\boldsymbol{\omega} = \dot{l}_i$$

We then omit the superscript A and we can rearrange the 6 equations into a matrix form

$$ \boxed{\dot{\boldsymbol{\mathcal{L}}} = \boldsymbol{J} \dot{\boldsymbol{\mathcal{X}}}} \tag{32}$$

Jacobian Matrix of a Stewart Platform

$$ \boldsymbol{J} = \begin{bmatrix} \hat{\boldsymbol{s}}_1^T & (\boldsymbol{b}_1 \times \hat{\boldsymbol{s}}_1)^T \\ \hat{\boldsymbol{s}}_2^T & (\boldsymbol{b}_2 \times \hat{\boldsymbol{s}}_2)^T \\ \hat{\boldsymbol{s}}_3^T & (\boldsymbol{b}_3 \times \hat{\boldsymbol{s}}_3)^T \\ \hat{\boldsymbol{s}}_4^T & (\boldsymbol{b}_4 \times \hat{\boldsymbol{s}}_4)^T \\ \hat{\boldsymbol{s}}_5^T & (\boldsymbol{b}_5 \times \hat{\boldsymbol{s}}_5)^T \\ \hat{\boldsymbol{s}}_6^T & (\boldsymbol{b}_6 \times \hat{\boldsymbol{s}}_6)^T \end{bmatrix} \tag{33}$$

$\boldsymbol{J}$ then **depends only** on:

- $\hat{\boldsymbol{s}}_i$ the orientation of the limbs
- $\boldsymbol{b}_i$ the position of the joints with respect to O_B and express in $\{\boldsymbol{A}\}$.

b Singularity Analysis

It is of primary importance to avoid singularities in a given workspace. To study the singularity configurations of the Stewart-Gough platform, we consider the Jacobian matrix determined with the equation (33).

From equation (26), it is clear that for the Stewart-Gough platform, $\boldsymbol{J}_q = \boldsymbol{I}$ and $\boldsymbol{J}_x = \boldsymbol{J}$. Hence the manipulator has **no inverse kinematic singularities** within the manipulator workspace, but **may possess forward kinematic singularity** when $\boldsymbol{J}$ becomes rank deficient. This may occur when

$$\det \boldsymbol{J} = 0$$

4.7 Static Forces in Parallel Manipulators

The relation between the forces/moments applied to the environment at the point of contact and the actuator forces/torques is determined and analyzed in the study of static force analysis.

It is assumed that the manipulator is at a static equilibrium, and that the actuator forces required to produce the desired contact forces are determined.

Two methods are usually applied: the **free-body diagram** and the **principle of virtual work**.

In the **free-body diagram approach**, the actuator forces are determined to produce desired contact forces/moments as well as the internal and interacting forces/torques applied at the limbs. The analysis of such forces is essential in the design of a manipulator to determine the stresses and deflection of each link and joint.

However, if only the actuator forces are desired to be determined, the **principle of virtual work** is more efficient and computationally less expensive.

a Virtual Work Approach

A virtual displacement for a parallel manipulator refers to an infinitesimal change in the general displacement of the moving platform as a result of any arbitrary infinitesimal changes in the joint variables at a given instant of time.

The virtual displacement of the joints can be written as $\delta \boldsymbol{q} = [\delta q_1, \delta q_2, \cdots, \delta q_m]^T$ and $\delta \boldsymbol{\mathcal{X}} = [\delta x, \delta y, \delta z, \delta \theta_x, \delta \theta_y, \delta \theta_z]^T$ denotes the virtual displacement of a contacting point of the moving platform. $[\delta \theta_x, \delta \theta_y, \delta \theta_z]^T = \delta \theta \hat{\boldsymbol{s}}$ are the orientation variables represented by screw coordinates.

Let the vector of actuator forces be denoted by $\boldsymbol{\tau} = [\tau_1, \tau_2, \cdots, \tau_m]^T$, and the external forces/torque acting on the contact point of the moving platform denoted by a wrench in a screw coordinate as $\boldsymbol{\mathcal{F}} = [\boldsymbol{f}, \boldsymbol{n}]^T$ in which $\boldsymbol{f} = [f_x, f_y, f_z]^T$ denotes the external forces, and $\boldsymbol{n} = [n_x, n_y, n_z]^T$ denotes the external torque action on the moving platform at the point of contact to the environment.

We assume that the frictional forces acting on the joints are negligible, and also that the gravitational forces of the limb links are much smaller than the interacting force of the moving platform to the environment. The principle of virtual work states that the total virtual work, δW, done by all actuators and external forces is equal to zero:

$$\boxed{\delta W = \boldsymbol{\tau}^T \delta \boldsymbol{q} - \boldsymbol{\mathcal{F}}^T \delta \boldsymbol{\mathcal{X}} = 0} \tag{34}$$

Furthermore, from the definition of the Jacobian, the virtual displacements $\delta \boldsymbol{q}$ and $\delta \boldsymbol{\mathcal{X}}$ are related by the Jacobian:

$$\delta \boldsymbol{q} = \boldsymbol{J} \cdot \delta \boldsymbol{\mathcal{X}}$$

We then have $\left(\boldsymbol{\tau}^T \boldsymbol{J} - \boldsymbol{\mathcal{F}}^T\right) \delta \boldsymbol{\mathcal{X}} = 0$ that holds for any arbitrary virtual displacement $\delta \boldsymbol{\mathcal{X}}$, hence

$$\boldsymbol{\tau}^T \boldsymbol{J} - \boldsymbol{\mathcal{F}}^T = 0$$

We obtain that the **Jacobian matrix** constructs the **transformation needed to find the actuator forces $\boldsymbol{\tau}$ from the wrench acting on the moving platform $\boldsymbol{\mathcal{F}}$**:

$$\boldsymbol{\mathcal{F}} = \boldsymbol{J}^T \boldsymbol{\tau} \tag{35}$$

b Static Forces of the Stewart-Gough Platform

As shown in Figure 8, the twist of moving platform is described by a 6D vector $\dot{\boldsymbol{\mathcal{X}}} = \left[{}^A\boldsymbol{v}_P \ {}^A\boldsymbol{\omega}\right]^T$, in which ${}^A\boldsymbol{v}_P$ is the velocity of point O_B, and ${}^A\boldsymbol{\omega}$ is the angular velocity of moving platform.

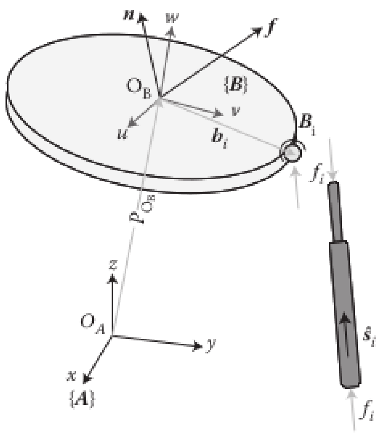

Figure 8 – *Free-body diagram of forces and moments action on the moving platform and each limb of the Stewart-Gough platform*

Consider an external wrench generated by the manipulator and applied to the environment at the point O_B denoted by $\boldsymbol{\mathcal{F}} = [\boldsymbol{f} \ \boldsymbol{n}]^T$.
It is assumed that no external forces are applied to the limbs except the actuator forces. Therefore, the static force can be assumed to be **along the limb axis $\hat{\boldsymbol{s}}_i$**, and the limb is subject to a tension/compression force f_i.
At static equilibrium, the summation of all acting forces on the moving platform shall be zero, therefore

$$-\boldsymbol{f} + \sum_{i=1}^{6} f_i \hat{\boldsymbol{s}}_i = 0 \tag{36}$$

in which $-\boldsymbol{f}$ if the **external force** applied to the moving platform from the environment.

The summation of moments contributed by all forces acting on the moving platform about O_B is as follows:

$$-\boldsymbol{n} + \sum_{i=1}^{6} b_i \times f_i \hat{\boldsymbol{s}}_i = 0 \tag{37}$$

in which $-n$ is the **external moment** applied to the moving platform by the environment, and b_i is the position vector from the point O_B to the attached point B_i on the moving platform.

Writing the two equations together in a matrix form results in

$$\begin{bmatrix} \hat{\boldsymbol{s}}_1 & \hat{\boldsymbol{s}}_2 & \cdots & \hat{\boldsymbol{s}}_6 \\ \boldsymbol{b}_1 \times \hat{\boldsymbol{s}}_1 & \boldsymbol{b}_2 \times \hat{\boldsymbol{s}}_2 & \cdots & \boldsymbol{b}_6 \times \hat{\boldsymbol{s}}_6 \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_6 \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{n} \end{bmatrix} \tag{38}$$

There we can recognize the transpose of the Jacobian matrix:

$$\boxed{\boldsymbol{\mathcal{F}} = \boldsymbol{J}^T \boldsymbol{\tau}} \tag{39}$$

in which $\boldsymbol{\tau} = [f_1, f_2, \cdots, f_6]^T$ is the vector of actuator forces, and $\boldsymbol{\mathcal{F}} = [\boldsymbol{f}, \boldsymbol{n}]^T$ is the 6D wrench applied by the manipulator to the environment.

4.8 Stiffness Analysis of Parallel Manipulators

Here, we focus on the deflections of the manipulator moving platform that are the result of the applied wrench to the environment. The amount of these deflections are a function of the applied wrench as well as the manipulator **structural stiffness**. Thus, the stiffness of a manipulator has a direct impact on its overall positioning accuracy if the manipulator is in contact with a stiff environment.

a Stiffness and Compliance Matrices

The relation between the applied actuator force τ_i and the corresponding small deflection Δq_i along the applied force axis can be approximated as a **linear function**:

$$\boxed{\tau_i = k_i \cdot \Delta q_i} \tag{40}$$

in which k_i denotes the **stiffness constant of the actuator**.

Re-writing the equation (40) for all limbs in a matrix form result in

$$\boxed{\boldsymbol{\tau} = \mathcal{K} \cdot \Delta \boldsymbol{q}} \tag{41}$$

in which $\boldsymbol{\tau}$ is the vector of actuator forces, and $\Delta \boldsymbol{q}$ corresponds to the actuator deflections. $\mathcal{K} = \text{diag}\,[k_1\ k_2 \dots k_m]$ is an $m \times m$ diagonal matrix composed of the actuator stiffness constants.

Writing the Jacobian relation given in equation (27) for infinitesimal deflection read

$$\Delta \boldsymbol{q} = \boldsymbol{J} \cdot \Delta \boldsymbol{\mathcal{X}} \tag{42}$$

in which $\Delta \boldsymbol{\mathcal{X}} = [\Delta x\ \Delta y\ \Delta z\ \Delta\theta x\ \Delta\theta y\ \Delta\theta z]$ is the infinitesimal linear and angular deflection of the moving platform.
Furthermore, rewriting the Jacobian as the projection of actuator forces to the moving platform (35) gives

$$\boldsymbol{\mathcal{F}} = \boldsymbol{J}^T \boldsymbol{\tau} \tag{43}$$

Hence, by substituting (41) and (42) in (43), we obtain:

$$\boxed{\boldsymbol{\mathcal{F}} = \underbrace{\boldsymbol{J}^T \mathcal{K} \boldsymbol{J}}_{\boldsymbol{K}} \cdot \Delta \boldsymbol{\mathcal{X}}} \tag{44}$$

Equation (44) implies that the moving platform output wrench is related to its deflection by the **stiffness matrix** K.

Stiffness Matrix

$$\boldsymbol{K} = \boldsymbol{J}^T \mathcal{K} \boldsymbol{J} \tag{45}$$

The stiffness matrix has desirable characteristics for analysis:

- It is a **symmetric positive definite matrix**, however, it is configuration dependent
- If the manipulator actuators have all the same stiffness constants k, the stiffness matrix is reduced to the form $\boldsymbol{K} = k \boldsymbol{J}^T \boldsymbol{J}$

If the stiffness matrix is inversible ($\det\big(\boldsymbol{J}^T \boldsymbol{J}\big) \neq 0$), the **compliance matrix** of the manipulator is defined as

$$\boxed{\boldsymbol{C} = \boldsymbol{K}^{-1} = (\boldsymbol{J}^T \mathcal{K} \boldsymbol{J})^{-1}} \tag{46}$$

The compliance matrix of a manipulator shows the mapping of the moving platform wrench to its deflection by

$$\Delta \boldsymbol{\mathcal{X}} = \boldsymbol{C} \cdot \boldsymbol{\mathcal{F}} \tag{47}$$

b Transformation Ellipsoid

As seen previously, the Jacobian matrix $\boldsymbol{J}$ transforms n-dimensional moving platform velocity vector $\dot{\boldsymbol{\mathcal{X}}}$ into m-dimensional actuated joint velocity $\dot{\boldsymbol{q}}$. Also, the Jacobian transpose $\boldsymbol{J}^T$ maps m-dimensional actuated joint forces $\boldsymbol{\tau}$ into n-dimensional applied wrench $\boldsymbol{\mathcal{F}}$.

One way to **characterize these transformation** is to compare the amplitude and direction of the moving platform velocity generated by a **unit** actuator joint velocity. To achieve this goal, we confine the actuator joint velocity vector on a m-dimensional unit sphere

$$\dot{\boldsymbol{q}}^T \dot{\boldsymbol{q}} = 1$$

and compare the resulting moving platform velocity in n-dimensional space:

$$\dot{\boldsymbol{\mathcal{X}}}^T \boldsymbol{J}^T \boldsymbol{J} \dot{\boldsymbol{\mathcal{X}}} = 1$$

Similarly, we can confine the exerted moving platform wrench $\boldsymbol{\mathcal{F}}^T \boldsymbol{\mathcal{F}} = 1$ and compare the required actuator forces: $\boldsymbol{\tau}^T \boldsymbol{J} \boldsymbol{J}^T \boldsymbol{\tau} = 1$.

Consider the case of fully parallel manipulators. Then $\boldsymbol{J}\boldsymbol{J}^T$ and $\boldsymbol{J}^T\boldsymbol{J}$ transformations are represented by $n \times n$ matrices. Geometrically, these transformations represent a **hyper-ellipsoid** in n-dimensional space, whose principal axes are the **eigenvectors** of $\boldsymbol{J}^T\boldsymbol{J}$ and $\boldsymbol{J}\boldsymbol{J}^T$ respectively. Furthermore, the lengths of the principal axes are equal to the reciprocals of the square roots of the eigenvalues of $\boldsymbol{J}\boldsymbol{J}^T$ and $\boldsymbol{J}^T\boldsymbol{J}$, which are also equal to the reciprocals of the **singular values** of $\boldsymbol{J}$.

The shape of this hyper-ellipsoid in space indicates the characteristics of the transformation. As this hyper-ellipsoid is closer to a hyper-sphere, the transformation becomes more uniform in different directions. Since Jacobian matrix is **configuration dependent**, the shape of the hyper-ellipsoid is also configuration dependent, and as the moving platform moves from one pose to the other, the shape of the hyper-ellipsoid changes accordingly.

A measure of the dexterity of the manipulator is the reciprocal of the Jacobian matrix condition number defined as:

$$\frac{1}{\kappa} = \frac{\sigma_{\min}}{\sigma_{\max}} \tag{48}$$

in which $\sigma_{\min}$ and $\sigma_{\max}$ are the smallest and the largest singular values of the Jacobian matrix.

c Stiffness Analysis of the Stewart-Gough Platform

In this section, we restrict our analysis to a 3-6 structure (Figure 9) in which there exist six distinct attachment points A_i on the fixed base and three moving attachment point B_i.
Denote the vector of actuated joint forces by $\boldsymbol{\tau} = [f_1\ f_2\ f_3\ f_4\ f_5\ f_6]$, and the corresponding vector of infinitesimal displacements of actuated limbs denoted by $\Delta\boldsymbol{L} = [\Delta l_1\ \Delta l_2\ \Delta l_3\ \Delta l_4\ \Delta l_5\ \Delta l_6]$. The relation between $\Delta\boldsymbol{L}$ and $\boldsymbol{\tau}$ is described by a diagonal 6×6 matrix $\mathcal{K}$:

$$\boldsymbol{\tau} = \mathcal{K} \cdot \Delta\boldsymbol{L}$$

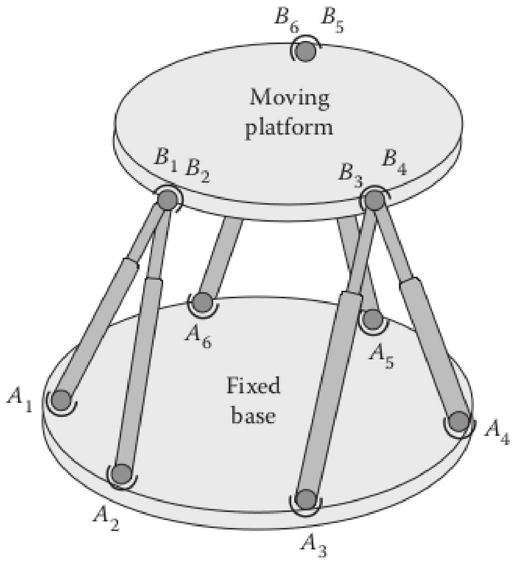

Figure 9 – *Schematic of a 3-6 Stewart-Gough platform*

Also, from the definition of the Jacobian, we have:

$$\Delta\boldsymbol{L} = \boldsymbol{J} \cdot \Delta\boldsymbol{\mathcal{X}}$$

in which $\Delta\boldsymbol{\mathcal{X}} = [\Delta_x\ \Delta_y\ \Delta_z\ \Delta\theta_x\ \Delta\theta_y\ \Delta\theta_z]$ is the vector of infinitesimal linear and angular motions of the moving platform.
Also, the vector of the moving platform output wrench denoted by $\boldsymbol{F} = [f_x\ f_y\ f_z\ n_x\ n_y\ n_z]$ is related to the vector of actuated joint forces $\boldsymbol{\tau}$ by:

$$\boldsymbol{F} = \boldsymbol{J}^T \cdot \boldsymbol{\tau}$$

By substitution, we obtain:

$$\boldsymbol{\mathcal{F}} = \boldsymbol{K} \cdot \Delta\boldsymbol{\mathcal{X}}$$

in which

$$\boldsymbol{K} = \boldsymbol{J}^T \mathcal{K} \boldsymbol{J}$$

where K is called the **stiffness matrix** of the Stewart-Gough manipulator.

For a given configuration of the moving platform, the eigenvalue of the stiffness matrix represents the stiffness of the manipulator in the corresponding eigenvector direction. Furthermore, the reciprocal of the stiffness matrix condition number may be used to represent the dexterity of the manipulator.

The maximum stiffness of the manipulator can be analyzed by the maximum singular value of the Jacobian matrix. The largest axis of the stiffness transformation hyper-ellipsoid is given by this value at each configuration.

5 Dynamics

5.1 Introduction

The dynamic analysis of parallel manipulators presents an inherent complexity due to their closed-loop structure. Several approaches have been proposed.

Traditional **Newton-Euler** formulation is used for dynamic analysis of general parallel manipulators. In this formulation, the equation of motion of each limb and the moving platform must be derived, which inevitably leads to a large number of equations and less computational efficiency. On the other hand, all the reaction forces can be computed, which is very useful in the design of a parallel manipulator.

The **Lagrangian** formulation eliminates all the unwanted reaction forces at the outset, and therefore, is quite efficient. However, because of the constraints imposed by the closed-loop structure, deriving explicit equations of motions in terms of a set of generalized coordinates becomes a prohibitive task.

A third approach is to use the **principle of virtual work**, in which the computation of the constraint forces are bypassed. In this method, inertial forces and moments are computed using linear and angular accelerations of the bodies. Then, the whole manipulator is considered to be in static equilibrium by using the d'Alembert's principle, and the principle of virtual work is applied to derive the input forces and torques.

Different objectives require different forms of formulations, there are three key issues pursued to derive dynamic formulation of parallel manipulators:

1. **Calculation of internal forces** either active or passive for the design process of the manipulator
2. **Study on dynamical properties** of the manipulator for controller design
3. Utilization of dynamic specifications in an inverse dynamics controller or any **model-based control topology**

The first item is the main advantage of the Newton-Euler formulation, the second and third items are the benefits of using the Lagrange or virtual work approaches.

The dynamic equations in an **explicit form** can be written as:

$$\boxed{\boldsymbol{M}(\boldsymbol{\mathcal{X}})\ddot{\boldsymbol{\mathcal{X}}} + \boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \boldsymbol{G}(\boldsymbol{\mathcal{X}}) = \boldsymbol{\mathcal{F}}} \tag{49}$$

in which:

- $\boldsymbol{\mathcal{X}}$ is a vector of the generalized coordinates
- $\boldsymbol{M}(\boldsymbol{\mathcal{X}})$ denotes the system **mass matrix**
- $\boldsymbol{C}(\boldsymbol{X}, \dot{\boldsymbol{X}})$ denotes the **Coriolis and centrifugal matrix**
- $\boldsymbol{G}(\boldsymbol{X})$ denotes the **gravity vector**
- $\boldsymbol{\mathcal{F}}$ denotes the generalized force

Deriving explicit dynamic equations for parallel manipulators is a very prohibitive task because of the closed-loop nature of the manipulator.

5.2 Dynamics of the Rigid Bodies

a Acceleration of Rigid Bodies

Acceleration Analysis - Definition

The acceleration analysis consists of studying the variations of linear velocity of a point and angular velocity of a rigid body with respect to time.

Direct differentiation of these vectors with respect to time in a **fixed** frame leads to linear velocity of a point and angular velocity of a rigid body, respectively. Note that, to determine the absolute linear velocity of a point, the derivative must be calculated relative to a fixed frame. In the study of robotic manipulators, usually **multiple moving frames** are defined to carefully determine the motion of the moving platform. Therefore, it is necessary to define the required arithmetics to transform the relative accelerations into absolute ones.

Angular Acceleration of a Rigid Body To define angular acceleration of a rigid body, consider a moving frame $\{\boldsymbol{B}\}$ attached to the rigid body, and the motion analyzed with respect to a fixed frame. Angular acceleration is an attribute of a rigid body and describes the variation of angular velocity of frame $\{\boldsymbol{B}\}$ with respect to time.

Angular Acceleration Vector

Angular acceleration vector, denoted by the symbol $\dot{\boldsymbol{\Omega}}$, describes the instantaneous change of the angular velocity of frame $\{\boldsymbol{B}\}$, denoted by $\boldsymbol{\Omega}$, with respect to the fixed frame $\{\boldsymbol{A}\}$:

$$\begin{aligned}
\dot{\boldsymbol{\Omega}} = \frac{d\boldsymbol{\Omega}}{dt} &= \ddot{\theta}\hat{\boldsymbol{s}} + \dot{\theta}\dot{\hat{\boldsymbol{s}}} \\
&= \ddot{\theta}\hat{\boldsymbol{s}} + \dot{\theta}(\boldsymbol{\Omega} \times \hat{\boldsymbol{s}}) \\
&= \ddot{\theta}\hat{\boldsymbol{s}}
\end{aligned} \tag{50}$$

where $\{\theta, \hat{\boldsymbol{s}}\}$ are the screw parameters representing the rotation of the rigid body.

As shown by (50), the angular acceleration of the rigid body is also along the screw axis $\hat{\boldsymbol{s}}$ with a magnitude equal to $\ddot{\theta}$.

Linear Acceleration of a Point Linear acceleration of a point P can be easily determined by time derivative of the velocity vector $\boldsymbol{v}_P$ of that point with respect to a fixed frame:

$$\boldsymbol{a}_p = \dot{\boldsymbol{v}}_p = \left(\frac{d\boldsymbol{v}_p}{dt}\right)_{\text{fix}} \tag{51}$$

Note that this is correct only if the derivative is taken with respect to a **fixed** frame.

Now consider the general motion of a rigid body, in which a moving frame $\{\boldsymbol{B}\}$ is attached to the rigid body and the problem is to find the absolute acceleration of point P with respect to the fixed frame $\{\boldsymbol{A}\}$. The rigid body performs a general motion, which is a combination of a translation, denoted by the velocity vector ${}^A\boldsymbol{v}_{O_B}$, and an instantaneous angular rotation denoted by $\boldsymbol{\Omega}$ (see Figure 7). To determine acceleration of point P, we start with the relation between absolute and relative velocities of point P:

$${}^A\boldsymbol{v}_P = {}^A\boldsymbol{v}_{O_B} + {}^A\boldsymbol{R}_B{}^B\boldsymbol{v}_P + {}^A\boldsymbol{\Omega}^{\times A}\boldsymbol{R}_B{}^B\boldsymbol{P} \tag{52}$$

In order to derive acceleration of point P, we differentiate both sides with respect to time and we obtain

$$\begin{aligned}
{}^A\boldsymbol{a}_p &= {}^A\boldsymbol{a}_{O_B} \quad \text{(linear acc. of } \{\boldsymbol{B}\}) \\
&+ {}^A\boldsymbol{R}_B{}^B\boldsymbol{a}_p \quad \text{(relative acc. of } P \text{ w.r.t. } \{\boldsymbol{B}\}) \\
&+ {}^A\dot{\boldsymbol{\Omega}}^{\times A}\boldsymbol{R}_B{}^B\boldsymbol{P} \quad \text{(angular acceleration of } \{\boldsymbol{B}\}) \\
&+ {}^A\boldsymbol{\Omega}^{\times}({}^A\boldsymbol{\Omega}^{\times A}\boldsymbol{R}_B{}^B\boldsymbol{P}) \quad \text{(centrifugal)} \\
&+ 2{}^A\boldsymbol{\Omega}^{\times A}\boldsymbol{R}_B{}^B\boldsymbol{v}_P \quad \text{(Coriolis)}
\end{aligned} \tag{53}$$

For the case where P is a point embedded in the rigid body, ${}^B\boldsymbol{v}_P = 0$ and ${}^B\boldsymbol{a}_P = 0$ and we obtain:

$$\begin{aligned}
{}^A\boldsymbol{a}_P = {}^A\boldsymbol{a}_{O_B} &+ {}^A\dot{\boldsymbol{\Omega}}^{\times A}\boldsymbol{R}_B{}^B\boldsymbol{P} \\
&+ {}^A\boldsymbol{\Omega}^{\times}({}^A\boldsymbol{\Omega}^{\times A}\boldsymbol{R}_B{}^B\boldsymbol{P})
\end{aligned} \tag{54}$$

b Mass Properties

In this section, the properties of mass, namely **center of mass**, **moments of inertia** and its characteristics and the required transformations are described.

Center of Mass Consider a reference frame $\{\boldsymbol{A}\}$ in which the mass distribution of a material body is measured, and let $\boldsymbol{p}$ denote the position vector of a differential mass ρdV with respect to a reference frame. The **center of mass** of a rigid body is defined as the point C which satisfied the following condition

$$\boldsymbol{p}_c = \frac{1}{m}\int_V \boldsymbol{p}\rho dV \tag{55}$$

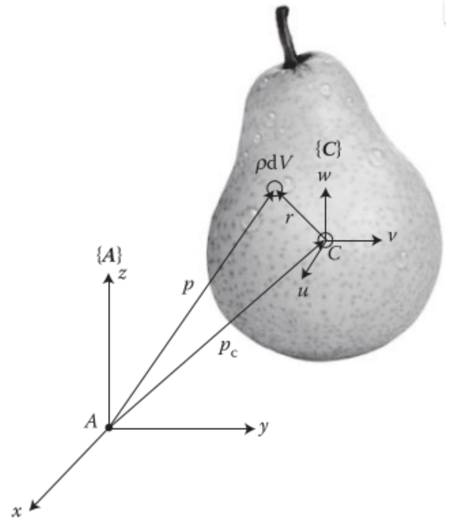

Figure 10 – *Mass properties of a rigid body*

in which the mass of the material body $\{\boldsymbol{B}\}$ with density ρ and volume V is defined as

$$m = \int_V \rho dV \tag{56}$$

Moments of Inertia As opposed to the mass, which introduces inertia to linear accelerations, moment of inertia is the property of mass which introduces **inertia to angular accelerations**. Basically, for rotational motion, **the distribution of mass with respect to the axis of rotation introduces resistance to the angular acceleration**.

Moments of inertia $\boldsymbol{I}$ about $\boldsymbol{A}$ is defined by the second moment of the mass with respect to a reference frame of rotation as:

$${}^A\boldsymbol{I} = \begin{bmatrix} I_{XX} & I_{XY} & I_{XZ} \\ I_{YX} & I_{YY} & I_{YZ} \\ I_{ZX} & I_{ZY} & I_{ZZ} \end{bmatrix} \tag{57}$$

in which

$$\begin{aligned}
I_{XX} &= \int_V (y^2+z^2)\rho dV, \quad I_{XY} = I_{YX} = -\int_V xy\rho dV \\
I_{YY} &= \int_V (x^2+z^2)\rho dV, \quad I_{YZ} = I_{ZY} = -\int_V yz\rho dV \\
I_{ZZ} &= \int_V (x^2+y^2)\rho dV, \quad I_{XZ} = I_{ZX} = -\int_V xz\rho dV
\end{aligned}$$

Principal Axes As seen in equation (57), the inertia matrix elements are a function of mass distribution of the rigid body with respect to the frame $\{\boldsymbol{A}\}$. Hence, it is possible to find **orientations of frame** $\{\boldsymbol{A}\}$ in which the product of inertia terms vanish and inertia matrix becomes **diagonal**:

$${}^A\boldsymbol{I} = \begin{bmatrix} I_{XX} & 0 & 0 \\ 0 & I_{YY} & 0 \\ 0 & 0 & I_{ZZ} \end{bmatrix} \tag{58}$$

Such axes are called the **principal axes of inertia**, and diagonal terms are called the **principal moments of inertia**, which represent the maximum, minimum and intermediate values of the moments of inertia for a particular chosen origin $\{\boldsymbol{A}\}$.
It can be shown that the principal moments of inertial and principal axes are invariant parameters and can be determined from an eigen value decomposition of the inertia matrix in any configuration of the reference frame $\{\boldsymbol{A}\}$.

Inertia Matrix Transformations The moment of inertia is usually given for **frames passing through the center of mass of the rigid body**. **The inertia matrix changes under change of the reference frame**.

Consider frame $\{\boldsymbol{C}\}$ **parallel** to $\{\boldsymbol{A}\}$ and attached to the center of mass of a rigid body and let $\boldsymbol{p}_c = [x_c, y_c, z_c]^T$ denote the vector of the position of the center of mass with respect to frame $\{\boldsymbol{A}\}$. The relation between the inertia matrix about A and that about C is given by the following relation:

$$\boxed{{}^A\boldsymbol{I} = {}^C\boldsymbol{I} + m(\boldsymbol{p}_c^T\boldsymbol{p}_c\boldsymbol{I}_{3\times3} - \boldsymbol{p}_c\boldsymbol{p}_c^T)} \tag{59}$$

in which m denotes the mass of the rigid body and $\boldsymbol{I}_{3\times3}$ denotes the identity matrix.

On the other hand, if the reference frame $\{B\}$ has **pure rotation** with respect to the frame attached to the center of mass $\{A\}$:

$$ {}^A\boldsymbol{I} = {}^A\boldsymbol{R}_C {}^C\boldsymbol{I} {}^A\boldsymbol{R}_C^T \tag{60}$$

c Momentum and Kinetic Energy

Linear Momentum Linear momentum of a material body, shown in Figure 11, with respect to a reference frame $\{\boldsymbol{A}\}$ is defined as

$$ {}^A\boldsymbol{G} = \int_V \frac{d\boldsymbol{p}}{dt}\rho dV \tag{61}$$

For any mass element ρdV, the position vector $\boldsymbol{p}$ can be written as

$$p = p_c + r$$

And because $\int_V r\rho dV = 0$, we have by substitution

$$ {}^A\boldsymbol{G} = \frac{d\boldsymbol{p}_c}{dt}\int_V \rho dV$$

and thus

$$\boxed{{}^A\boldsymbol{G} = m \cdot {}^A\boldsymbol{v}_C} \tag{62}$$

in which ${}^A\boldsymbol{v}_C$ denotes the velocity of the center of mass with respect to the frame $\{\boldsymbol{A}\}$.

This result implies that the **total linear momentum** of differential masses is equal to the linear momentum of a **point mass** m located at the **center of mass**. This highlights the important of the center of mass in dynamic formulation of rigid bodies.

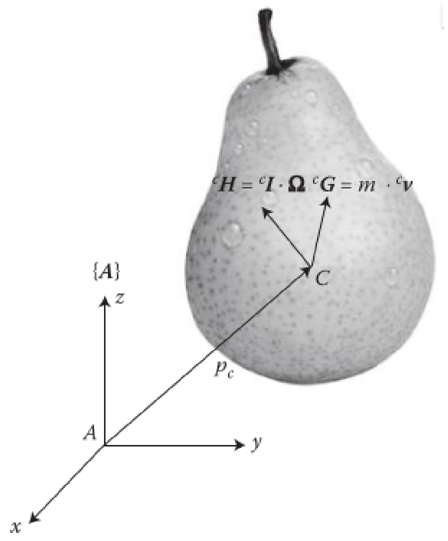

Figure 11 – *The components of the angular momentum of a rigid body about A*

Angular Momentum Consider the solid body represented in Figure 11. Angular momentum of the differential masses ρdV about a reference point A, expressed in the reference frame $\{\boldsymbol{A}\}$ is defined as

$$ {}^A\boldsymbol{H} = \int_V \left(\boldsymbol{p} \times \frac{d\boldsymbol{p}}{dt}\right)\rho dV$$

in which $d\boldsymbol{p}/dt$ denotes the velocity of differential mass with respect to the reference frame $\{\boldsymbol{A}\}$.
By substituting $\boldsymbol{p} = \boldsymbol{p}_c + \boldsymbol{r}$ in the previous equations, be obtain:

$$ {}^A\boldsymbol{H} = \boldsymbol{p}_c \times m\boldsymbol{v}_c + \int_V \boldsymbol{r} \times (\boldsymbol{\Omega} \times \boldsymbol{r})\rho dV$$

Therefore, angular momentum of the rigid body about point A is reduced to

$$\boxed{{}^A\boldsymbol{H} = \boldsymbol{p}_c \times \boldsymbol{G}_c + {}^C\boldsymbol{H}} \tag{63}$$

in which

$$ {}^C\boldsymbol{H} = \int_V \boldsymbol{r} \times (\boldsymbol{\Omega} \times \boldsymbol{r})\rho dV = {}^C\boldsymbol{I} \cdot \boldsymbol{\Omega}$$

Equation (63) reveals that angular momentum of a rigid body about a point A can be written as $\boldsymbol{p}_c \times \boldsymbol{G}_c$, which is the contribution of linear momentum of the rigid body about point A, and ${}^C\boldsymbol{H}$ which is the angular momentum of the rigid body about the center of mass. This also highlights the important of the center of mass in the dynamic analysis of rigid bodies. If the center of mass is taken as the reference point, the relation describing angular momentum (63) is very analogous to that of linear momentum (62).

Kinetic Energy The Kinetic energy of a rigid body is defined as

$$\boxed{\boldsymbol{K} = \frac{1}{2}\int_V \boldsymbol{v}\cdot\boldsymbol{v}\rho dV} \tag{64}$$

The velocity of a differential mass ρdV can be represented by linear velocity of the center of mass and angular velocity of the rigid body as

$$\boldsymbol{v} = \boldsymbol{v}_p + \boldsymbol{\Omega}\times\boldsymbol{r}$$

By substitution, the kinetic energy of the rigid body may be obtained by:

$$\boxed{\boldsymbol{K} = \frac{1}{2}\boldsymbol{v}_c\times\boldsymbol{G}_c + \frac{1}{2}\boldsymbol{\Omega}\cdot{}^C\boldsymbol{H}} \tag{65}$$

in which $\boldsymbol{G}_C$ is the linear momentum of the rigid body and ${}^C\boldsymbol{H}$ is the angular momentum of the rigid body about the center of mass.

This equation reveals that kinetic energy of a moving body can be represented as the **kinetic energy of a point mass located as the center of mass**, in addition to the **kinetic energy of a body rotating about the center of mass**.

d Newton-Euler Laws

The Newton and Euler laws can be written for three different cases where the angular motion:

1. is about a fixed point in space
2. is represented about the center of mass
3. is represented about an arbitrary moving point in space

We only examine the case in which all rotations are represented about the center of mass.

Consider a rigid body under general motion, that is, a combination of translation and rotation.

Newtown's law

The Newton's law relates the change of linear momentum of the rigid body to the resulting external forces applied to it

$$\sum \boldsymbol{f}_{\text{ext}} = \frac{d\boldsymbol{G}_c}{dt}$$

For the case of a constant mass rigid body, this law is reduced to

$$\sum \boldsymbol{f}_{\text{ext}} = m\frac{d\boldsymbol{v}_c}{dt} = m\boldsymbol{a}_c$$

in which $\boldsymbol{a}_c$ is the linear acceleration of the center of mass.

Euler's law

The Euler's law relates the change of angular momentum of a rigid body about the center of mass, to the summation of all external moments applied to the rigid body about center of mass

$$\sum {}^c\boldsymbol{n}_{\text{ext}} = \frac{d}{dt}({}^c\boldsymbol{H})$$

For the case of a constant mass rigid body, this law is reduced to

$$\sum {}^c\boldsymbol{n}_{\text{ext}} = \frac{d}{dt}({}^c\boldsymbol{I}\boldsymbol{\Omega}) = {}^c\boldsymbol{I}\dot{\boldsymbol{\Omega}} + \boldsymbol{\Omega}\times({}^c\boldsymbol{I}\boldsymbol{\Omega})$$

in which $\sum {}^c\boldsymbol{n}_{\text{ext}}$ is the summation of all external moments applied to the rigid body about the center of mass, ${}^c\boldsymbol{I}$ is the moment of inertia about the center of mass, and $\boldsymbol{\Omega}$ is the angular velocity of the rigid body.

5.3 Newton-Euler Formulation

The most popular approach used in robotics to derive the dynamic equation of motion of a parallel manipulator is the **Newton-Euler formulation**.
In the Newton-Euler formulation, the **free-body diagrams** of all the limbs and moving platform are considered and the **Newton Euler laws are applied to each isolated body**. To apply the laws to each body, it is necessary to derive linear acceleration of links, center of mass, as well as angular acceleration of the links. Hence, **acceleration analysis** would be performed on all the links of the manipulator and the moving platform.

Furthermore, all the external forces and moments applied to the links and to the moving platform must be carefully determined. Gravitational forces acting on the center of masses, frictional forces and moments acting on the joints, and any possible disturbance force or moment applied to the links and to the moving platform would be identified. The most important external forces or moments applied on the manipulator are the one applied by the actuators, denoted by $\boldsymbol{\tau} = [\tau_1, \tau_2, \dots, \tau_m]^T$. The forces and moments shall be derived from the set of Newton-Euler laws, which are written separately for each link and the moving platform.

Finally, by elimination of these constraints forces and moments on the Newton-Euler equations written for the moving platform, the dynamic equations **relating the actuator forces and moments $\boldsymbol{\tau}$ to the motion variables of the moving platform $\boldsymbol{\mathcal{X}}$, $\dot{\boldsymbol{\mathcal{X}}}$** and **$\ddot{\boldsymbol{\mathcal{X}}}$** are derived.

a Dynamic Formulation of the Stewart-Gough Platform

Acceleration Analysis In acceleration analysis, it is intended to **derive expressions for linear and angular acceleration of the limbs**, namely $\ddot{l}_i$ and $\dot{\boldsymbol{\omega}}_i$ as a function of the moving platform acceleration $\ddot{\boldsymbol{\mathcal{X}}} = [\dot{\boldsymbol{v}}_p, \dot{\boldsymbol{\omega}}]^T$. To obtain such a relation, let us rewrite the **velocity loop closure**:

$$\boldsymbol{v}_p + \boldsymbol{\omega} \times \boldsymbol{b}_i = \dot{l}_i \hat{\boldsymbol{s}}_i + l_i(\boldsymbol{\omega}_i \times \hat{\boldsymbol{s}}_i) \tag{66}$$

Since there is no actuation torque about $\hat{\boldsymbol{s}}_i$, the limb angular velocity and acceleration vectors ($\boldsymbol{\omega}_i$ and $\dot{\boldsymbol{\omega}}_i$) are normal to $\hat{\boldsymbol{s}}_i$ provided that the following assumption are considered for the platform:

- both end joints of the limb are spherical
- the limbs are symmetric with respect to their axes
- the effects of friction in spherical joints are neglected

Considering these assumptions, it can be concluded that the limbs cannot spin about their axes: $\boldsymbol{\omega}_i \cdot \hat{\boldsymbol{s}}_i = 0$ and $(\hat{\boldsymbol{s}}_i \times (\boldsymbol{\omega}_i \times \hat{\boldsymbol{s}}_i)) = \boldsymbol{\omega}_i$.
To obtain the angular velocity of the limbs $\boldsymbol{\omega}_i$, we cross multiply $\hat{\boldsymbol{s}}_i$ to both sides of the previous equation:

$$\boldsymbol{\omega}_i = \frac{1}{l_i}(\hat{\boldsymbol{s}}_i \times \boldsymbol{v}_{b_i}) \tag{67}$$

With $\boldsymbol{v}_{b_i}$ an **intermediate variable** corresponding to the velocity of point $\boldsymbol{b}_i$:

$$\boldsymbol{v}_{b_i} = \boldsymbol{v}_p + \boldsymbol{\omega} \times \boldsymbol{b}_i \tag{68}$$

As illustrated in Figure 12, the piston-cylinder structure of the limbs is decomposed into two separate parts, the masses of which are denoted by m_{i_1} and m_{i_2}. The position vector of these two center of masses can be determined by the following equations:

$$\boldsymbol{p}_{i_1} = \boldsymbol{a}_i + c_{i_1} \hat{\boldsymbol{s}}_i \tag{69}$$

$$\boldsymbol{p}_{i_2} = \boldsymbol{a}_i + (l_i - c_{i_2}) \hat{\boldsymbol{s}}_i \tag{70}$$

By differentiating the previous equations and doing some manipulations, we obtain:

$$\ddot{l}_i = \boldsymbol{a}_{b_i} \times \hat{\boldsymbol{s}}_i + l_i(\boldsymbol{\omega_i} \cdot \boldsymbol{\omega_i}) \tag{71}$$

$$\dot{\boldsymbol{\omega}}_i = \frac{1}{l_i}(\hat{\boldsymbol{s}}_i \times \boldsymbol{a}_{b_i} - 2\dot{l}_i \boldsymbol{\omega}_i) \tag{72}$$

$$\boldsymbol{a}_{i_1} = c_{i_1}(\dot{\boldsymbol{\omega}}_i \times \hat{\boldsymbol{s}}_i + \boldsymbol{\omega}_i \times (\boldsymbol{\omega}_i \times \hat{\boldsymbol{s}}_i)) \tag{73}$$

$$\boldsymbol{a}_{i_2} = (l_i - c_{i_2})(\dot{\boldsymbol{\omega}}_i \times \hat{\boldsymbol{s}}_i - (\boldsymbol{\omega}_i \cdot \boldsymbol{\omega}_i)\hat{\boldsymbol{s}}_i) + 2\dot{l}_i(\boldsymbol{\omega}_i \times \hat{\boldsymbol{s}}_i) + \ddot{l}_i \hat{\boldsymbol{s}}_i \tag{74}$$

with

$$\boldsymbol{a}_{b_i} = \boldsymbol{a}_p + \dot{\boldsymbol{\omega}} \times \boldsymbol{b}_i + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \boldsymbol{b}_i) \tag{75}$$

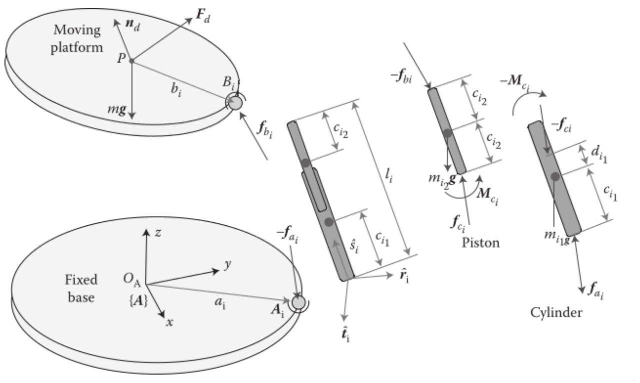

Figure 12 – *Free-body diagram of the limbs and the moving platform of a general Stewart-Gough manipulator*

Dynamic Formulation of the Limbs To derive the dynamic formulation of the Stewart-Gough platform, the manipulator is decomposed into a moving platform and six identical limbs. We assume that each limb consists of two parts, the cylinder and the piston, where the velocities and the accelerations of their centers of masses are determined. We also assume that the centers of masses of the cylinder and the piston are located at a distance of c_{i_1} and c_{i_2} above their foot points, and their masses are denoted by m_{i_1} and m_{i_2}. Moreover, consider that the pistons are symmetric about their axes, and their centers of masses lie at their midlengths.
The free-body diagrams of the limbs and the moving platforms is given in Figure 12. The reaction forces at fixed points A_i are denoted by $\boldsymbol{f}_{a_i}$, the internal force at moving points B_i are dentoed by $\boldsymbol{f}_{b_i}$, and the internal forces and moments between cylinders and pistons are denoted by $\boldsymbol{f}_{c_i}$ and $\boldsymbol{M}_{\boldsymbol{c_i}}$ respectively.
Assume that the only existing external disturbance wrench is applied on the moving platform and is denoted by $\boldsymbol{\mathcal{F}}_d = [\boldsymbol{F}_d, \boldsymbol{n}_d]^T$.

$$\begin{aligned} \boldsymbol{f}_{b_i} = & \frac{1}{l_i}(I_{xx_i} + l_i^2 m_{c_e})\hat{\boldsymbol{s}}_i \times \dot{\boldsymbol{\omega}}_i \\ & + \frac{2}{l_i} m_{i_2} c_{i_2} \dot{l}_i \hat{\boldsymbol{s}}_i \times \boldsymbol{\omega}_i \\ & - m_{g_e} \hat{\boldsymbol{s}}_i \times (\hat{\boldsymbol{s}}_i \times \boldsymbol{g}) \end{aligned} \tag{76}$$

in which m_{c_e} is defined as

$$m_{c_e} = \frac{1}{l_i^2}\left(m_{i_1} c_{i_1}^2 + m_{i_2} c_{i_2}^2\right) \tag{77}$$

Dynamic Formulation of the Moving Platform Assume that the **moving platform center of mass is located at the center point** P and it has a mass m and moment of inertia ${}^A\boldsymbol{I}_P$. Furthermore, consider that gravitational force and external disturbance wrench are applied on the moving platform, $\boldsymbol{\mathcal{F}}_d = [\boldsymbol{F}_d, \boldsymbol{n}_d]^T$ as depicted in Figure 12.
The Newton-Euler formulation of the moving platform

is as follows:

$$\sum \boldsymbol{F}_{\text{ext}} = \sum_{i=1}^{6} \boldsymbol{f}_{b_i} + m\boldsymbol{g} + \boldsymbol{F}_d = m\boldsymbol{a}_p \tag{78}$$

$$\begin{aligned}\sum {}^{p}\boldsymbol{n}_{\text{ext}} &= \boldsymbol{n}_d + \sum_{i=1}^{6} \boldsymbol{b}_i \times \boldsymbol{f}_{b_i} \\ &= {}^{A}\boldsymbol{I}_P\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times {}^{A}\boldsymbol{I}_P\boldsymbol{\omega}\end{aligned} \tag{79}$$

in which ${}^{A}\boldsymbol{I}_P$ is considered in the fixed frame $\{\boldsymbol{A}\}$ and can be calculated by:

$${}^{A}\boldsymbol{I}_P = {}^{A}\boldsymbol{R}_B {}^{B}\boldsymbol{I}_P {}^{A}\boldsymbol{R}_B^T \tag{80}$$

These equations can be rewritten in an implicit form as

Dynamic Formulation - Stewart Platform

$$m(\boldsymbol{a}_p - \boldsymbol{g}) - \boldsymbol{F}_d - \sum_{i=1}^{6} \boldsymbol{f}_{b_i} = \boldsymbol{0} \tag{81}$$

$${}^{A}\boldsymbol{I}_P\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times {}^{A}\boldsymbol{I}_P\boldsymbol{\omega} - \boldsymbol{n}_d - \sum_{i=1}^{6} \boldsymbol{b}_i \times \boldsymbol{f}_{b_i} = \boldsymbol{0} \tag{82}$$

These two equations are the governing dynamic formulation of the Stewart-Gough platform, in which $\boldsymbol{\mathcal{F}}_d = [\boldsymbol{F}_d, \boldsymbol{n}_d]^T$ denotes the disturbance wrench exerted on the moving plateform.
They can be viewed in an implicit vector form of

$$\boldsymbol{f}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}}, \ddot{\boldsymbol{\mathcal{X}}}, \boldsymbol{\mathcal{F}}_d, \boldsymbol{\tau}) = \boldsymbol{0} \tag{83}$$

in which $\boldsymbol{\mathcal{X}} = [\boldsymbol{x}_P, \boldsymbol{\theta}]^T$ is the motion variable of the moving platform consisting of the linear position of point P and the moving platform orientation represented by screw coordinates.

b Closed-Form Dynamics

While dynamic formulation in the form of Equation (83) can be used to simulate inverse dynamics of the Stewart-Gough platform, its implicit nature makes it unpleasant for the dynamic analysis and control.

Closed-Form Dynamics of the Limbs To derive a closed-form dynamic formulation for the Stewart-Gough platform as

$$\boldsymbol{M}(\boldsymbol{\mathcal{X}})\ddot{\boldsymbol{\mathcal{X}}} + \boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \boldsymbol{G}(\boldsymbol{\mathcal{X}}) = \boldsymbol{\mathcal{F}} \tag{84}$$

first consider an intermediate generalized coordinate x_i, which is in fact the position of point b_i. This generalized coordinate is used to harmonize the limb and the moving platform dynamic formulation and to **derive an closed-form structure for the whole manipulator**.
Now, manipulate each limb dynamic equations to convert them into the closed form. Let us first introduce some relations to substitute kinematic parameters like $\dot{\boldsymbol{\omega}}_i$, $\ddot{\boldsymbol{\omega}}_i$, $\dot{l}_i$ with the intermediate generalized coordinate x_i and its time derivatives.
After some manipulations, we obtain the following closed form equation:

$$\boldsymbol{M}_i\ddot{\boldsymbol{x}}_i + \boldsymbol{C}_i\dot{\boldsymbol{x}}_i + \boldsymbol{G}_i = \boldsymbol{F}_i \tag{85}$$

the corresponding mass matrix $\boldsymbol{M}_i$, the Coriolis matrix $\boldsymbol{C}_i$, and the gravity vector $\boldsymbol{G}_i$ can be simplified into the following form:

$$\boldsymbol{M}_i = m_{i_2}\hat{\boldsymbol{s}}_i\hat{\boldsymbol{s}}_i^T - \frac{1}{l_i^2} I_{xx_i}\hat{\boldsymbol{s}}_{i\times}^2 \tag{86}$$

$$\boldsymbol{C}_i = -\frac{2}{l_i} m_{c_o}\dot{l}_i\hat{\boldsymbol{s}}_{i\times}^2 - \frac{1}{l_i^2} m_{i_2}c_{i_2}\hat{\boldsymbol{s}}_i\dot{\boldsymbol{x}}_i^T\hat{\boldsymbol{s}}_{i\times}^2 \tag{87}$$

$$\boldsymbol{G}_i = \left(m_{g_e}\hat{\boldsymbol{s}}_{i\times}^2 - m_{i_2}\hat{\boldsymbol{s}}_i\hat{\boldsymbol{s}}_i^T\right)\boldsymbol{g} \tag{88}$$

$$\boldsymbol{F}_i = -\boldsymbol{f}_{b_i} + \tau_i\hat{\boldsymbol{s}}_i \tag{89}$$

in which

$$m_{c_e} = \frac{1}{l_i^2}(m_{i_1}c_{i_1}^2 + m_{i_2}c_{i_2}^2) \tag{90}$$

$$m_{c_o} = \frac{1}{l_i} m_{i_2}c_{i_2} - \frac{1}{l_i^2}(I_{xx_i} + l_i^2 m_{c_e}) \tag{91}$$

$$m_{g_e} = \frac{1}{l_i}(m_{i_1}c_{i_1} + m_{i_2}(l_i - c_{i_2})) \tag{92}$$

Closed-Form Dynamics of the Moving Platform In this section, the dynamic equations of the moving platform are transformed in the following closed-form formulation

$$\boldsymbol{M}_p\ddot{\boldsymbol{\mathcal{X}}} + \boldsymbol{C}_p\dot{\boldsymbol{\mathcal{X}}} + \boldsymbol{G}_p = \boldsymbol{\mathcal{F}}_p \tag{93}$$

in which $\boldsymbol{\mathcal{X}}$ consists of six coordinates: the first three $\boldsymbol{x}_p$ represent linear motion of the moving platform, and the last three $\boldsymbol{\theta}$ its angular motion. It is preferable to use the **screw coordinates** for representing the angular motion **as its derivative is also a vector representing angular velocity**:

$$\boxed{\boldsymbol{\mathcal{X}} = \begin{bmatrix}\boldsymbol{x}_p \\ \boldsymbol{\theta}\end{bmatrix}; \quad \dot{\boldsymbol{\mathcal{X}}} = \begin{bmatrix}\boldsymbol{v}_p \\ \boldsymbol{\omega}\end{bmatrix}; \quad \ddot{\boldsymbol{\mathcal{X}}} = \begin{bmatrix}\boldsymbol{a}_p \\ \dot{\boldsymbol{\omega}}\end{bmatrix}} \tag{94}$$

Equations (81) and (82) can be simply converted into a closed form of Equation (93) with the following terms:

$$\begin{aligned}\boldsymbol{M}_p &= \begin{bmatrix} m\boldsymbol{I}_{3\times3} & \boldsymbol{O}_{3\times3} \\ \boldsymbol{O}_{3\times3} & {}^{A}\boldsymbol{I}_p \end{bmatrix}_{6\times6}; \boldsymbol{C}_p = \begin{bmatrix} \boldsymbol{O}_{3\times3} & \boldsymbol{O}_{3\times3} \\ \boldsymbol{O}_{3\times3} & \omega_{\times}{}^{A}\boldsymbol{I}_p \end{bmatrix}_{6\times6} \\ \boldsymbol{G}_p &= \begin{bmatrix} -m\boldsymbol{g} \\ \boldsymbol{O}_{3\times1} \end{bmatrix}_{6\times1}; \boldsymbol{\mathcal{F}}_p = \begin{bmatrix} \boldsymbol{F}_d + \sum \boldsymbol{f}_{b_i} \\ \boldsymbol{n}_d + \sum \boldsymbol{b}_{i\times}\boldsymbol{f}_{b_i} \end{bmatrix}_{6\times1}\end{aligned} \tag{95}$$

Closed-Form Dynamics of the Stewart-Gough Manipulator To derive the closed-form dynamic formulation for the whole manipulator, a transformation is required to map the intermediate generalized coordinates x_i into the principal generalized coordinates $\boldsymbol{\mathcal{X}}$.

Using such a transformation, and by adding the resulting equations of the limbs and the moving platform, the internal forces $\boldsymbol{f}_{b_i}$ can be eliminated, and closed-form dynamic formulation for the whole manipulator can be derived.

To generate such a transformation define a Jacobian matrix $\boldsymbol{J}_i$ relating the intermediate coordinates to that of the principal generalized coordinate:

$$\dot{\boldsymbol{x}}_i = \boldsymbol{J}_i \dot{\boldsymbol{\mathcal{X}}} \tag{96}$$

in which

$$\boldsymbol{J}_i = \begin{bmatrix} \boldsymbol{I}_{3\times 3} & -\boldsymbol{b}_{i\times} \end{bmatrix} \tag{97}$$

$$\boldsymbol{M}_{li}\ddot{\boldsymbol{x}}_i + \boldsymbol{C}_{li} + \dot{\boldsymbol{x}}_i + \boldsymbol{G}_{li} = \boldsymbol{\mathcal{F}}_{li} \tag{98}$$

in which

$$\begin{aligned} \boldsymbol{M}_{li} = \boldsymbol{J}_i^T \boldsymbol{M}_i \boldsymbol{J}_i; \quad \boldsymbol{C}_{li} = J_i^T \boldsymbol{M}_i \dot{\boldsymbol{J}}_i + \boldsymbol{J}_i^T \boldsymbol{C}_i \boldsymbol{J}_i \\ \boldsymbol{G}_{li} = \boldsymbol{J}_i^T G_i; \quad \boldsymbol{\mathcal{F}}_{li} = \boldsymbol{J}_i^T \boldsymbol{F}_i \end{aligned} \tag{99}$$

$$\begin{aligned} \boldsymbol{M}(\boldsymbol{\mathcal{X}}) &= \boldsymbol{M}_p + \sum_{i=1}^{6} \boldsymbol{M}_{li} \\ \boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}}) &= \boldsymbol{C}_p + \sum_{i=1}^{6} \boldsymbol{C}_{li} \\ \boldsymbol{G}(\boldsymbol{\mathcal{X}}) &= \boldsymbol{G}_p + \sum_{i=1}^{6} \boldsymbol{G}_{li} \\ \boldsymbol{\mathcal{F}}(\boldsymbol{\mathcal{X}}) &= \boldsymbol{\mathcal{F}}_d + \sum_{i=1}^{6} \boldsymbol{\mathcal{F}}_{\tau_i} \end{aligned} \tag{100}$$

Forward Dynamics Simulations As shown in Figure 13, it is **assumed that actuator forces and external disturbance wrench applied to the manipulator are given and the resulting trajectory of the moving platform is to be determined**.

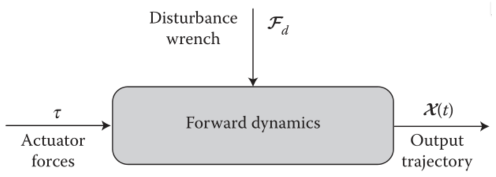

Figure 13 – *Flowchart of forward dynamics implementation sequence*

The closed-form dynamic formulation of the Stewart-Gough platform corresponds to the set of equations given in (84), whose terms are given in (95).

Inverse Dynamics Simulation In inverse dynamics simulations, it is assumed that the **trajectory of the manipulator is given**, and the **actuator forces required to generate such trajectories are to be determined**.

As illustrated in Figure 14, inverse dynamic formulation is implemented in the following sequence. The first step is trajectory generation for the manipulator moving platform. Many different algorithms are developed for a smooth trajectory generation. For such a trajectory, $\boldsymbol{\mathcal{X}}_d(t)$ and the time derivatives $\dot{\boldsymbol{\mathcal{X}}}_d(t)$, $\ddot{\boldsymbol{\mathcal{X}}}_d(t)$ are known.

The next step is to solve the inverse kinematics of the manipulator and to find the limbs' linear and angular positions, velocity and acceleration as a function of the manipulator trajectory. The manipulator Jacobian matrix $\boldsymbol{J}$ is also calculated in this step.

Next, the dynamic matrices given in the closed-form formulations of the limbs and the moving platform are calculated using equations (86) and (95), respectively.

To combine the corresponding matrices, an to generate the whole manipulator dynamics, it is necessary to find intermediate Jacobian matrices $\boldsymbol{J}_i$, given in (97), and then compute compatible matrices for the limbs given in (99). Now that all the terms required to **computed to actuator forces required to generate such a trajectory** is computed, let us define $\boldsymbol{\mathcal{F}}$ as the resulting Cartesian wrench applied to the moving platform. This wrench can be calculated from the summation of all inertial and external forces **excluding the actuator torques $\boldsymbol{\tau}$** in the closed-form dynamic formulation (84). By this definition, $\boldsymbol{\mathcal{F}}$ can be viewed as the projector of the actuator forces acting on the manipulator, mapped to the Cartesian space. Since there is no redundancy in actuation in the Stewart-Gough manipulator, the Jacobian matrix $\boldsymbol{J}$, squared and actuator forces can be uniquely determined from this wrench, by $\boldsymbol{\tau} = \boldsymbol{J}^{-T}\boldsymbol{\mathcal{F}}$, provided $\boldsymbol{J}$ is non-singular. Therefore, actuator forces $\boldsymbol{\tau}$ are computed in the simulation from

$$\boldsymbol{\tau} = \boldsymbol{J}^{-T}\left(\boldsymbol{M}(\boldsymbol{\mathcal{X}})\ddot{\boldsymbol{\mathcal{X}}} + \boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \boldsymbol{G}(\boldsymbol{\mathcal{X}}) - \boldsymbol{\mathcal{F}}_d\right) \tag{101}$$

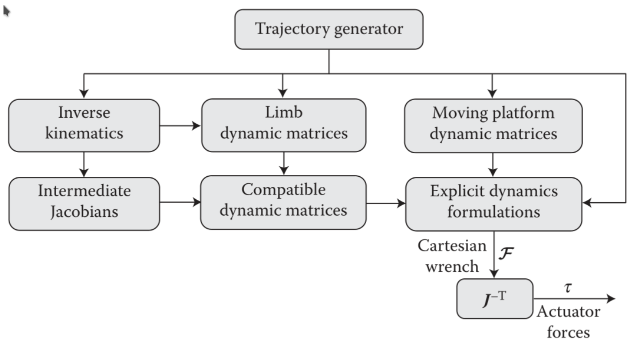

Figure 14 – *Flowchart of inverse dynamics implementation sequence*

5.4 **TODO** Virtual Work Formulation

5.5 **TODO** Lagrange Formulation

$$K(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}}) = \frac{1}{2}\dot{\boldsymbol{\mathcal{X}}}^T \boldsymbol{M}(\boldsymbol{\mathcal{X}})\dot{\boldsymbol{\mathcal{X}}} \tag{102}$$

$$\boldsymbol{G}(\boldsymbol{\mathcal{X}}) = \frac{\partial P(\boldsymbol{\mathcal{X}})}{\partial \boldsymbol{\mathcal{X}}} \tag{103}$$

$$\boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}}) = \frac{1}{2}(\dot{\boldsymbol{M}} + \boldsymbol{U}^T - \boldsymbol{U}) \tag{104}$$

6 Motion Control

6.1 Introduction

Parallel robots are designed for two different types of applications.

In the first type, the moving platform of the robot accurately follows a **desired position and orientation** path in a specific time frame, while **no interacting forces** need to be applied to the environment.

The second type of application include situations where the robot moving platform is in **contact with a stiff environment** (e.g. precision machining). In such application, the contact force describe the state of interaction more effectively than the position and orientation of the moving platform. The problem of **force control** can be described as to derive the actuator forces for such a manipulator required to generate a prescribed desired wrench (force and torque) at the manipulator moving platform, while the manipulator is performing its motion.

Although a multiple degrees-of-freedom robotic manipulator can usually be represented by a MIMO and nonlinear model, many industrial controllers for such robots consist of a number of linear controller designed to **control individual joint motions**. One of the reasons why such decentralization can perform well in practice is the use of large gear reductions in robot actuators, which significantly reduces the coupling and non linear behavior of robot dynamics.

However, using advanced techniques in nonlinear and MIMO control permits to overcome limitations of the SISO approach.

6.2 Controller Topology

Motion Control

In motion control of parallel manipulator, it is assumed that the controller computes the **required actuator forces** or torques to cause the robot motion to follow a desired position and orientation trajectory.

Let us use the motion variables as the generalized coordinate of the moving platform defined by $\boldsymbol{\mathcal{X}} = [\boldsymbol{x}_P, \boldsymbol{\theta}]^T$, in which the linear motion is represented by $\boldsymbol{x}_p = [x_p, y_p, z_p]^T$, while the moving platform orientation is represented by **screw coordinates** $\boldsymbol{\theta} = \theta[s_x, s_y, s_z]^T = [\theta_x, \theta_y, \theta_z]^T$.

Consider the general closed-form dynamics formulation of a parallel robot

$$\boxed{\boldsymbol{M}(\boldsymbol{\mathcal{X}})\ddot{\boldsymbol{\mathcal{X}}} + \boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \boldsymbol{G}(\boldsymbol{\mathcal{X}}) = \boldsymbol{\mathcal{F}}} \tag{105}$$

where

- $\boldsymbol{M}(\boldsymbol{\mathcal{X}})$ denotes the mass matrix
- $\boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})$ denotes the Coriolis and centrifugal matrix
- $\boldsymbol{G}(\boldsymbol{\mathcal{X}})$ denotes the gravity vector
- $\boldsymbol{\mathcal{F}}$ denotes the generalized forces applied to the moving platform center of mass

The generalized forces can be decomposed as follow

$$\boldsymbol{\mathcal{F}} = \boldsymbol{J}^T \boldsymbol{\tau} + \boldsymbol{\mathcal{F}}_d$$

with

- $\boldsymbol{J}$ is the Jacobian
- $\boldsymbol{\tau}$ are the actuator forces
- $\boldsymbol{\mathcal{F}}_d$ are any external wrenches

Control Topology - Definition

Control topology is referred to the **structure of the control system** used to compute the actuator forces/torques from the measurements, and the required pre and post processing.

For motion control of a manipulator, the controller has to compute the actuator force/torques required to cause the motion of the moving platform according to the desired trajectory. In general, the desired motion of the moving platform may be represented by the desired generalized coordinate of the manipulator, denoted by $\boldsymbol{\mathcal{X}}_d$.

To perform such motion in closed loop, it is necessary to **measure the output motion** $\boldsymbol{\mathcal{X}}$ of the manipulator by an instrumentation system. Such instrumentation usually consists of two subsystems: the first subsystem may use accurate accelerometers, or global positioning systems to calculate the position of a point on the moving platform; and a second subsystem may use inertial or laser gyros to determine orientation of the moving platform.

Figure 15 shows the general topology of a motion controller using direct measurement of the motion variable $\boldsymbol{\mathcal{X}}$, as feedback in the closed-loop system. In such a structure, the measured position and orientation of the manipulator is compared to its desired value to generate the **motion error vector** $\boldsymbol{e}_\mathcal{X}$. The controller uses this error information to generate

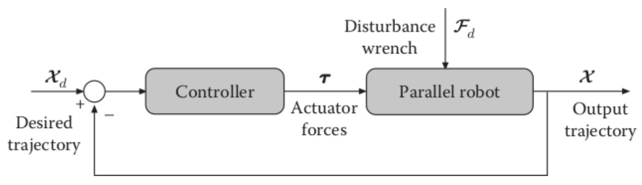

Figure 15 – *The general topology of motion feedback control: motion variable $\mathcal{X}$ is measured*

suitable commands for the actuators to minimize the tracking error.

However, it is usually much **easier to measure the active joint variable** rather than measuring the final position and orientation of the moving platform. The relation between the **joint variable q** and **motion variable** of the moving platform $\mathcal{X}$ is dealt with the **forward and inverse kinematics**. The relation between the **differential motion variables $\dot{q}$** and **$\dot{\mathcal{X}}$** is studied through the **Jacobian analysis**.

It is then possible to use the forward kinematic analysis to calculate $\mathcal{X}$ from the measured joint variables q, and one may use the control topology depicted in Figure 16 to implement such a controller.

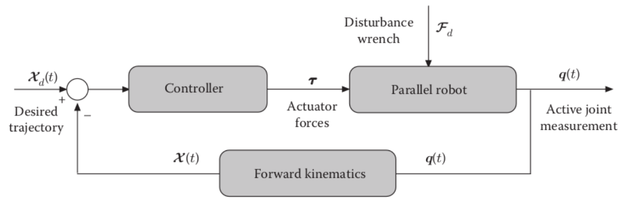

Figure 16 – *The general topology of motion feedback control: the active joint variable q is measured*

In this topology, the forward kinematic analysis of the manipulator has to be performed to implement the feedback loop. As described earlier, this is a **complex task** for parallel manipulators. It is even more complex when a solution has to be found in real time.

However, as shown herein before, the inverse kinematic analysis of parallel manipulators is much easier to carry out. To overcome the implementation problem of the control topology in Figure 16, another control topology is usually implemented for parallel manipulators.
In this topology, depicted in Figure 17, the desired motion trajectory of the robot $\mathcal{X}_d$ is used in an **inverse kinematic analysis** to find the corresponding desired values for joint variable q_d. Hence, the controller is designed based on the **joint space error e_q**.
Therefore, the **structure and characteristics** of the controller in this topology is totally **different** from that given in the first two topologies.
The **input and output** of the controller depicted in Figure 17 are **both in the joint space**. However, this is not the case in the previous topologies where the input to the controller is the motion error in task space, while its output is in the joint space.
For the topology in Figure 17, **independent controllers** for each joint may be suitable.

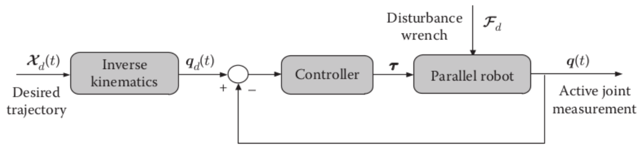

Figure 17 – *The general topology of motion feedback control: the active joint variable q is measured, and the inverse kinematic analysis is used*

To generate a **direct input to output relation in the task space**, consider the topology depicted in Figure 18. A force distribution block is added which maps the generated wrench in the task space $\mathcal{F}$, to its corresponding actuator forces/torque τ.

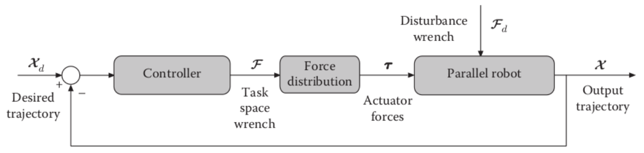

Figure 18 – *The general topology of motion feedback control in task space: the motion variable $\mathcal{X}$ is measured, and the controller output generates wrench in task space*

For a fully parallel manipulator such as the Stewart-Gough platform, this mapping can be constructed from the **Jacobian** transpose of the manipulator:

$$\mathcal{F} = J^T \tau; \quad \tau = J^{-T} \mathcal{F}$$

6.3 Motion Control in Task Space

a Decentralized PD Control

In the control structure in Figure 19, a number of linear PD controllers are used in a feedback structure on each error component. The decentralized controller consists of **six disjoint linear controllers** acting on each error component $e_x = [e_x, \ e_y, \ e_z, \ e_{\theta_x}, \ e_{\theta_y}, \ e_{\theta_z}]$. The PD controller is denoted by $K_d s + K_p$, in which K_d and K_p are 6×6 **diagonal matrices** denoting the derivative and proportional controller gains for each error term.

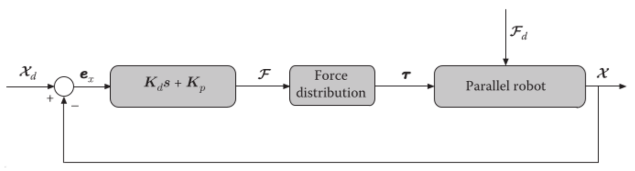

Figure 19 – *Decentralized PD controller implemented in task space*

Hence, by this structure, each tracking error component is **treated separately**. The output of the controller is denoted by $\boldsymbol{\mathcal{F}} = [F_x \ F_y \ F_z \ \tau_x \ \tau_y \ \tau_z]$.

In practice, the calculated output wrench is transformed into actuator forces through the force distribution block. This mapping is implemented through inverse of the manipulator **Jacobian** transpose by $\boldsymbol{\tau} = \boldsymbol{J}^{-T}\boldsymbol{\mathcal{F}}$.

Different alternatives of linear controllers can be used instead of the PD controller used in this structure, however PD controller is the simplest form which can preserve the manipulator stability while providing suitable tracking performance.

The proposed decentralized PD controller is very simple in structure and therefore easily implementable. The design of such a controller needs no detailed information on the manipulator dynamics. The controller gains are generally tuned experimentally based on physical realization of the controller by trial and error.

b Feed Forward Control

A feedforward wrench denoted by $\boldsymbol{\mathcal{F}}_{ff}$ may be added to the decentralized PD controller structure as depicted in Figure 20. This term is generated from the dynamic model of the manipulator in the task space, represented in a closed form by the following equation:

$$\boldsymbol{\mathcal{F}}_{ff} = \hat{\boldsymbol{M}}(\boldsymbol{\mathcal{X}}_d)\ddot{\boldsymbol{\mathcal{X}}}_d + \hat{\boldsymbol{C}}(\boldsymbol{\mathcal{X}}_d, \dot{\boldsymbol{\mathcal{X}}}_d)\dot{\boldsymbol{\mathcal{X}}}_d + \hat{\boldsymbol{G}}(\boldsymbol{\mathcal{X}}_d)$$

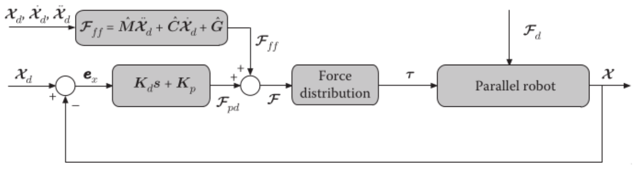

Figure 20 – *Feed forward wrench added to the decentralized PD controller in task space*

The desired trajectory in task space $\boldsymbol{\mathcal{X}}_d$, and its derivatives $\dot{\boldsymbol{\mathcal{X}}}_d$, $\ddot{\boldsymbol{\mathcal{X}}}_d$ are the required inputs for the feedforward block. This term is called feedforward since no online information of the output motion trajectory $\boldsymbol{\mathcal{X}}$ is needed for its computation.

In order to generate this term, dynamic formulation of the robots and its kinematic and dynamic parameters are needed. In practice, exact knowledge of dynamic matrices are not available, and therefore, **estimate** of these matrices are used in practice, denoted by $\hat{\boldsymbol{M}}$, $\hat{\boldsymbol{C}}$ and $\hat{\boldsymbol{G}}$.

The information required to generate the feedforward wrench $\boldsymbol{\mathcal{F}}_{ff}$ is usually available beforehand and can be derived offline. The closed-loop dynamic formulation for the manipulator becomes:

$$\begin{aligned}
\boldsymbol{M}(\boldsymbol{\mathcal{X}})\ddot{\boldsymbol{\mathcal{X}}} &+ \boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \boldsymbol{G}(\boldsymbol{\mathcal{X}}) \\
&= \boldsymbol{\mathcal{F}} + \boldsymbol{\mathcal{F}}_d \\
&= \boldsymbol{\mathcal{F}}_{pd} + \boldsymbol{\mathcal{F}}_{ff} + \boldsymbol{\mathcal{F}}_d \\
&= \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x + \boldsymbol{\mathcal{F}}_{ff} + \hat{\boldsymbol{M}}\ddot{\boldsymbol{\mathcal{X}}}_d + \hat{\boldsymbol{C}}\dot{\boldsymbol{\mathcal{X}}}_d + \hat{\boldsymbol{G}}
\end{aligned} \tag{106}$$

If the knowledge of the dynamic matrices is complete, we may assume that $\hat{\boldsymbol{M}} = \boldsymbol{M}$, $\hat{\boldsymbol{C}} = \boldsymbol{C}$ and $\hat{\boldsymbol{G}} = \boldsymbol{G}$. Furthermore, if we consider that the controller performs well such that $\boldsymbol{\mathcal{X}}(t) \simeq \boldsymbol{\mathcal{X}}_d(t)$ and $\dot{\boldsymbol{\mathcal{X}}}(t) \simeq \dot{\boldsymbol{\mathcal{X}}}_d(t)$, the simplified closed-loop dynamics become:

$$\begin{aligned}
\boldsymbol{M}(\ddot{\boldsymbol{\mathcal{X}}}_d - \ddot{\boldsymbol{\mathcal{X}}}) + \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x + \boldsymbol{\mathcal{F}}_d = 0 \\
\boldsymbol{M}\ddot{\boldsymbol{e}}_x + \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x + \boldsymbol{\mathcal{F}}_d = 0
\end{aligned} \tag{107}$$

This equation implies that, if the mentioned assumptions hold, the **error dynamics** satisfies a set of **second-order system** in the presence of disturbance. By choosing appropriate gains for PD controller, the transient and steady-state performance of tracking error can be designed so as to satisfy the application requirements.

Note that except the mass matrix, the error dynamic terms are all **configuration independent**, and therefore, it is much **easier to tune the PD controller gains** to work well within the whole workspace of the robot.

However, this method faces a number of **limitations** in practice. The most important limitation of this control technique is the **stringent assumption of a complete knowledge requirement of the dynamic matrices**. In practice, derivation of these matrices is a prohibitive task.

Finally, because of the dependency of the mass matrix to the configuration of the robot, the error dynamics are not completely decoupled. This means that correction in one error component may be considered as a disturbance effect to the other components. To overcome these limitations, inverse dynamic approach is given in the following section.

c Inverse Dynamics Control

Inverse Dynamics Control

In inverse dynamics control (IDC), nonlinear dynamics of the model is used to add a **corrective term** to the decentralized PD controller. By this means, **nonlinear and coupling behavior of the robotic manipulator is significantly attenuated**, and therefore, the performance of linear controller is greatly improved.

General structure of IDC applied to a parallel manipulator is depicted in Figure 21. A corrective wrench $\boldsymbol{\mathcal{F}}_{fl}$

is added in a **feedback structure** to the closed-loop system, which is calculated from the Coriolis and centrifugal matrix and gravity vector of the manipulator dynamic formulation.
Furthermore, mass matrix is added in the forward path in addition to the desired trajectory acceleration $\ddot{\boldsymbol{\mathcal{X}}}_d$.
As for the feedforward control, the **dynamics and kinematic parameters of the robot are needed**, and in practice estimates of these matrices are used.

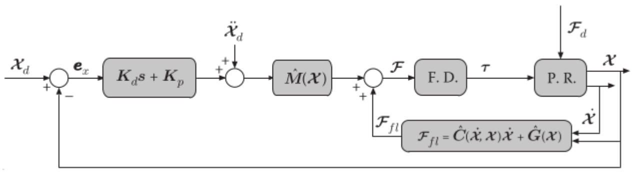

Figure 21 – *General configuration of inverse dynamics control implemented in task space*

The controller output wrench applied to the manipulator may be derived as follows:

$$\begin{aligned} \boldsymbol{\mathcal{F}} &= \hat{\boldsymbol{M}}(\boldsymbol{\mathcal{X}})\boldsymbol{a} + \boldsymbol{\mathcal{F}}_{fl} && \text{(108a)} \\ &= \hat{\boldsymbol{M}}(\boldsymbol{\mathcal{X}})\boldsymbol{a} + \hat{\boldsymbol{C}}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \hat{\boldsymbol{G}}(\boldsymbol{\mathcal{X}}) && \text{(108b)} \\ \boldsymbol{a} &= \ddot{\boldsymbol{\mathcal{X}}}_d + \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x && \text{(108c)} \end{aligned}$$

The closed-loop dynamic formulation for the manipulator becomes:

$$\begin{aligned} &\boldsymbol{M}(\boldsymbol{\mathcal{X}})\ddot{\boldsymbol{\mathcal{X}}} + \boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \boldsymbol{G}(\boldsymbol{\mathcal{X}}) \\ &\quad = \boldsymbol{\mathcal{F}} + \boldsymbol{\mathcal{F}}_d \\ &\quad = \hat{\boldsymbol{M}}(\boldsymbol{\mathcal{X}})\left(\ddot{\boldsymbol{\mathcal{X}}}_d + \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x\right) \\ &\qquad + \hat{\boldsymbol{C}}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \hat{\boldsymbol{G}}(\boldsymbol{\mathcal{X}}) + \boldsymbol{\mathcal{F}}_d \end{aligned} \qquad (109)$$

If the knowledge of the dynamic matrices is complete, the closed-loop dynamic formulation simplifies to:

$$\hat{\boldsymbol{M}}(\boldsymbol{\mathcal{X}})\left(\ddot{e}_d + \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x\right) + \boldsymbol{\mathcal{F}}_d = 0 \qquad (110)$$

This control technique is very popular in practice because of the fact that this technique can significantly **linearize and decouple dynamic formulation of the closed-loop error dynamics**. Furthermore, the error dynamic terms are all **configuration independent**, and therefore, it is much easier to tune the PD controller gains for suitable performance in the whole workspace of the robot.

However, note that for a good performance, and **accurate model of the system is required**, and the overall procedure is **not robust to modeling uncertainty**. Furthermore, this technique is computationally intensive in terms of the online computations needed to carry out the closed-loop control structure.

d Partial Linearization IDC

Inverse dynamics control has several features making it very attractive in practice. However, to apply this method, complete knowledge of the dynamic formulation matrices is required. This requirement has the main drawbacks that the dynamic formulation of the parallel manipulator is a complicated step to be carried out.

To implement all the terms in IDC structure, not only the structure and components of such matrices must be carefully determined, but also the kinematics and inertial parameters of the robot are needed to be identified and calibrated. This step requires the use of high-precision calibration equipment which are not usually accessible. Finally, if all the terms and parameters are well known, implementation of full inverse dynamic linearization is computationally intensive.
These are the reasons why, in practice, IDC control is extended to different forms where the above-mentioned stringent requirements are reduced.

To develop the simplest possible implementable IDC, let us recall dynamic formulation complexities:

- the manipulator mass matrix $\boldsymbol{M}(\boldsymbol{\mathcal{X}})$ is derived from kinetic energy of the manipulator (Eq. (102))
- the gravity vector $\boldsymbol{G}(\boldsymbol{\mathcal{X}})$ is derived from potential energy (Eq. (103))
- the Coriolis and centrifugal matrix $\boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})$ is derived from Eq. (103)

The computation of the Coriolis and centrifugal matrix is more intensive than that of the mass matrix. Gravity vector is more easily computable.
However, it is shown that certain properties hold for mass matrix, gravity vector and Coriolis and centrifugal matrix, which might be directly used in the control techniques developed for parallel manipulators. One of the most important properties of dynamic matrices is the skew-symmetric property of the matrix $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$.

Consider dynamic formulation of parallel robot given in Eq. (105), in which the skew-symmetric property of dynamic matrices is satisfied. The simplest form of IDC control effort $\boldsymbol{\mathcal{F}}$ consists of:

$$\boldsymbol{\mathcal{F}} = \boldsymbol{\mathcal{F}}_{pd} + \boldsymbol{\mathcal{F}}_{fl}$$

in which the first term $\boldsymbol{\mathcal{F}}_{pd}$ is generated by the simplified PD form on the motion error:

$$\boldsymbol{\mathcal{F}}_{pd} = \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p + \boldsymbol{e}_x$$

The second term $\boldsymbol{\mathcal{F}}_{fl}$ is considered to be only the gravity vector of the manipulator $\boldsymbol{G}(\boldsymbol{\mathcal{X}})$, at any configuration, and the computationally intensive Coriolis and centrifugal term is not used:

$$\boldsymbol{\mathcal{F}}_{fl} = \boldsymbol{G}(\boldsymbol{\mathcal{X}})$$

Note that for an appreciable tracking performance with no static error at steady state, it is required to have complete knowledge of **only** the gravity term. By this means, computations required in this control technique are significantly less than that of the general IDC.

Despite the simple structure of such a controller, the resulting control technique is very well performed, especially at steady state. We can show that this control topology achieves asymptotic tracking for a constant desired trajectory motion, that is, $\dot{\boldsymbol{\mathcal{X}}}_d = 0$.

This reveals the fact that even if the mass matrix and Coriolis and centrifugal matrix are not used in the feedback, and the closed-loop dynamics is not completely linearized, the PD control structure with gravity compensation can still lead to asymptotic tracking. However, to suitable transient performance, more information of the system dynamics must be used in the linearization technique given in IDC.

6.4 TODO Robust and Adaptative Control

Inverse dynamics control faces the stringent requirement that for a good performance, an **accurate model** of the system is required, and the overall procedure is **not robust** to modeling uncertainty. Furthermore, this technique is computationally intensive in terms of online computation needed to carry out the closed-loop control structure. The proposed modified inverse dynamics control, while being beneficial in terms of computational cost, is not suitable in terms of a closed-loop transient performance.

Another approach to modify IDC is to consider a complete linearization, but **assume that complete knowledge of dynamic formulation matrices is not available**. To **compensate for the lack of knowledge**, two advanced control methods, namely **robust** and **adaptive control** are proposed:

- In the **robust approach**, a **fixed controller** is designed to satisfy the control objectives for the **worst possible case of modeling uncertainty** and disturbance wrenches.
- In the **adaptive approach**, the estimates of dynamic formulation matrices are **updated** such that the difference between the true values of these matrices to their estimates converges to zero.

A global understanding of the trade-offs involved in each method is needed to employ either of them in practice.

a Robust Inverse Dynamics Control

Various sources of uncertainties such as unmodelled dynamics, unknown parameters, calibration error, unknown disturbance wrenches, and varying payloads may exist, and are not seen in dynamic model of the manipulator.

To consider these modeling uncertainty in the closed-loop performance of the manipulator, recall the general closed-form dynamic formulation of the manipulator given in Eq. (105), and modify the inverse dynamics control input $\boldsymbol{\mathcal{F}}$ as

$$\begin{aligned} \boldsymbol{\mathcal{F}} &= \hat{\boldsymbol{M}}(\boldsymbol{\mathcal{X}})\boldsymbol{a}_r + \hat{\boldsymbol{C}}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \hat{\boldsymbol{G}}(\boldsymbol{\mathcal{X}}) \\ \boldsymbol{a}_r &= \ddot{\boldsymbol{\mathcal{X}}}_d + \boldsymbol{K}_d \dot{\boldsymbol{e}}_x + \boldsymbol{K}_p \boldsymbol{e}_x + \boldsymbol{\delta}_a \end{aligned}$$

in which $\boldsymbol{a}_r$ is the robustified control input.
Comparing this equation to the usual IDC, a robustifying term $\boldsymbol{\delta}_a$ is added to compensate for modeling uncertainties.
Note that, as defined earlier, the notation $\hat{(.)}$ represents the estimated value of $(.)$ and $\tilde{(.)}$ is defined as the error mismatch between the estimated value and the true value as $\tilde{(.)} = \hat{(.)} - (.)$.
In a similar manner $\tilde{(.)}$ notation may be applied to the motion variables as

$$\tilde{\boldsymbol{\mathcal{X}}} = \boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{X}}_d = -\boldsymbol{e}_x$$

The closed-loop dynamic formulation of the manipulator can be written as:

$$\ddot{\boldsymbol{\mathcal{X}}} = \boldsymbol{a}_r + \boldsymbol{\eta}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}}, \boldsymbol{a}_r)$$

in which

$$\boldsymbol{\eta} = \boldsymbol{M}^{-1}\left(\tilde{\boldsymbol{M}}\boldsymbol{a}_r + \tilde{\boldsymbol{C}}\dot{\boldsymbol{\mathcal{X}}} + \tilde{\boldsymbol{G}}\right)$$

is a measure of modeling uncertainty.

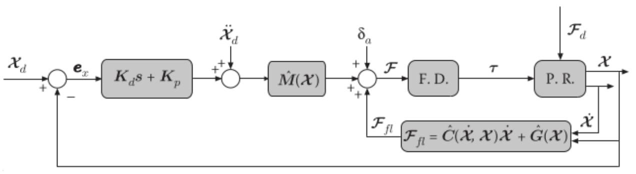

Figure 22 – *General configuration of robust inverse dynamics control implemented in the task space*

b Adaptive Inverse Dynamics Control

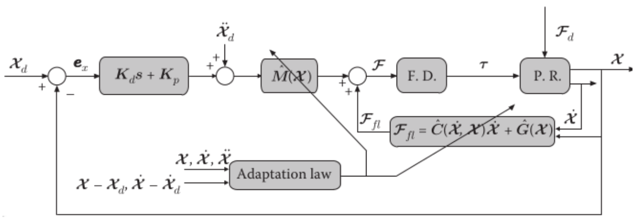
$\mathcal{X}_d$
e_x
$K_d s + K_p$
$\ddot{\mathcal{X}}_d$
$\hat{M}(\mathcal{X})$
$\mathcal{F}$
F. D.
τ
$\mathcal{F}_d$
P. R.
$\mathcal{X}$
$\dot{\mathcal{X}}$
$\mathcal{F}_{fl}$
$\mathcal{F}_{fl} = \hat{C}(\dot{\mathcal{X}}, \mathcal{X})\dot{\mathcal{X}} + \hat{G}(\mathcal{X})$
$\mathcal{X}, \dot{\mathcal{X}}, \ddot{\mathcal{X}}$
$\mathcal{X} - \mathcal{X}_d, \dot{\mathcal{X}} - \dot{\mathcal{X}}_d$
Adaptation law

Figure 23 – *General configuration of adaptative inverse dynamics control implemented in task space*

6.5 Motion Control in Joint Space

Although the motion control schemes developed in section 6.3 are very effective for tracking performance, they suffer from an implementation constraint that the motion variable $\boldsymbol{\mathcal{X}}$ must be measured in practice.

If this measurement is available without any doubt, such topologies are among the best routines to be implemented in practice. However, as explained in Section 6.2, in many practical situations measurement of the motion variable $\boldsymbol{\mathcal{X}}$ is difficult or expensive, and usually just the active joint variables $\boldsymbol{q}$ are measured. In such cases, the controllers developed in the joint space may be recommended for practical implementation.

To generate a direct input to output relation in the joint space, consider the topology depicted in Figure 16. In this topology, the controller input is the joint variable error vector $\boldsymbol{e}_q = \boldsymbol{q}_d - \boldsymbol{q}$, and the controller output is directly the actuator force vector $\boldsymbol{\tau}$, and hence there exists a **one-to-one correspondence between the controller input to its output**.

The general form of dynamic formulation of parallel robot is usually given in the task space. For motion control in joint space, we need to transform the dynamic formulation in the joint space, by which the actuator forces $\boldsymbol{\tau}$ are directly related to the active joint variables $\boldsymbol{q}$.

a Dynamic Formation in the Joint Space

The relation between the task space variables to their counterparts in the joint space can be derived by forward and inverse kinematics relations. Although both analyses involve solution to a set of non-linear equations, for parallel manipulators, inverse kinematic solution proves to be much easier to obtain than that of forward kinematic solution.

This relation in **differential kinematics** is much simpler and can be completely determined by the Jacobian matrix:

$$\boxed{\dot{q} = \boldsymbol{J}\dot{\boldsymbol{\mathcal{X}}} \Longrightarrow \dot{\boldsymbol{\mathcal{X}}} = \boldsymbol{J}^{-1}\dot{\boldsymbol{q}}}$$

The acceleration variables are then:

$$\ddot{q} = \dot{\boldsymbol{J}}\dot{\boldsymbol{\mathcal{X}}} + \boldsymbol{J}\ddot{\mathcal{X}} \Longrightarrow \ddot{X} = \boldsymbol{J}^{-1}\ddot{q} - \boldsymbol{J}^{-1}\dot{\boldsymbol{J}}\dot{\boldsymbol{\mathcal{X}}}$$

Furthermore, the relation between the actuator force vector $\boldsymbol{\tau}$ to the corresponding task space wrench is given by:

$$\boxed{\boldsymbol{\mathcal{F}} = \boldsymbol{J}^T\boldsymbol{\tau} \Longrightarrow \boldsymbol{\tau} = \boldsymbol{J}^{-T}\boldsymbol{\mathcal{F}}}$$

Substituting $\dot{\boldsymbol{\mathcal{X}}}$ and $\ddot{\boldsymbol{\mathcal{X}}}$ from the above equations into the dynamic formulation of the parallel robot gives:

$$\begin{aligned}&\left(\boldsymbol{J}^{-T}\boldsymbol{M}\boldsymbol{J}^{-1}\right)\ddot{\boldsymbol{q}}\\&\quad+\boldsymbol{J}^{-T}\left(\boldsymbol{C}-\boldsymbol{M}\boldsymbol{J}^{-1}\dot{\boldsymbol{J}}\right)\boldsymbol{J}^{-1}\dot{\boldsymbol{q}}\\&\quad+\boldsymbol{J}^{-T}\boldsymbol{G}+\boldsymbol{J}^{-T}\boldsymbol{\mathcal{F}}_d=\boldsymbol{\tau}\end{aligned}$$

Dynamic Formulation in the Joint Space

$$\boldsymbol{M}_q\ddot{\boldsymbol{q}} + \boldsymbol{C}_q\dot{\boldsymbol{q}} + \boldsymbol{G}_q + \boldsymbol{\tau}_d = \boldsymbol{\tau} \qquad (111)$$

with:

$$\begin{aligned}\boldsymbol{M}_q &= \boldsymbol{J}^{-T}\boldsymbol{M}\boldsymbol{J}^{-1} && (112a)\\ \boldsymbol{D}_q &= \boldsymbol{J}^{-T}\left(\boldsymbol{C}-\boldsymbol{M}\boldsymbol{J}^{-1}\dot{\boldsymbol{J}}\right)\boldsymbol{J}^{-1} && (112b)\\ \boldsymbol{G}_q &= \boldsymbol{J}^{-T}\boldsymbol{G} && (112c)\\ \boldsymbol{\tau}_q &= \boldsymbol{J}^{-T}\boldsymbol{\mathcal{F}}_d && (112d)\end{aligned}$$

Equation 111 represents the closed form dynamic formulation of a general parallel robot in the joint space.

Note that the dynamic matrices are **not** explicitly represented in terms of the joint variable vector $\boldsymbol{q}$. In fact, to fully derive these matrices, the Jacobian matrices must be computed and are generally derived as a function of the motion variables $\boldsymbol{\mathcal{X}}$. Furthermore, the main dynamic matrices are all functions of the motion variable $\boldsymbol{\mathcal{X}}$. Hence, in practice, to find the dynamic matrices represented in the joint space, **forward kinematics** should be solved to find the motion variable $\boldsymbol{\mathcal{X}}$ for any given joint motion vector $\boldsymbol{q}$.

Since in parallel robots the forward kinematic analysis is computationally intensive, there exist inherent difficulties in finding the dynamic matrices in the joint space as an explicit function of $\boldsymbol{q}$. In this case it is possible to solve forward kinematics in an online manner, it is recommended to use the control topology depicted in 16, and implement control law design in the task space.

However, one implementable alternative to calculate the dynamic matrices represented in the joint space is to use the **desired motion trajectory** $\boldsymbol{\mathcal{X}}_d$ instead of the true value of motion vector $\boldsymbol{\mathcal{X}}$ in the calculations. This approximation significantly reduces the computational cost, with the penalty of having mismatch between the estimated values of these matrices to their true values.

b Decentralized PD Control

The first control strategy introduced in the joint space consists of the simplest form of feedback control in such manipulators. In this control structure, depicted in Figure 24, a number of PD controllers are used in a feedback structure on each error component.

The PD controller is denoted by $\boldsymbol{K}_d s + \boldsymbol{K}_p$, where $\boldsymbol{K}_d$ and $\boldsymbol{K}_p$ are $n \times n$ **diagonal** matrices denoting the derivative and proportional controller gains, respectively.

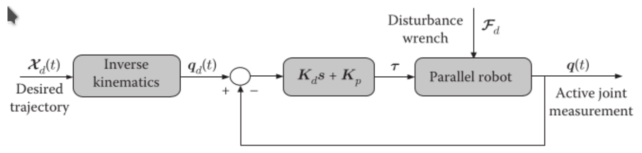

Figure 24 – *Decentralized PD controller implemented in joint space*

By this structure, each tracking error component is **treated separately** by its disjoint PD controller. The proposed decentralized PD controller is very simple in structure, and therefore very easy to be implemented on the manipulator. The design of such a controller **needs no detailed information on the manipulator dynamic formulation and parameters**. However, the tracking performance of such a controller is relatively poor, and **static tracking errors** might be unavoidable. Also, the performance of the closed-loop system is configuration dependent.

In practice, the gains are tuned experimentally and obtained as a trade-off between transient behavior and steady-state errors at different configurations. As the dynamics of the system in the joint space is configuration dependent, finding suitable controller gains to result in required performance in all configurations is a difficult task.

The performance of the controller to attenuate measurement noise and external disturbance wrenches are also poor in practice. To remedy these shortcomings, some modifications have been proposed to this structure and further described.

c Feedforward Control

The tracking performance of the simple PD controller implemented in the joint space is usually not sufficient at different configurations. To improve the tracking performance, a feedforward actuator force denoted by $\boldsymbol{\tau}_{ff}$ may be added to the structure of the controller as depicted in Figure 25.

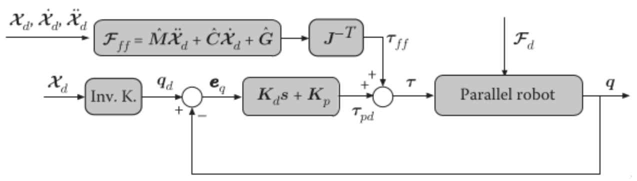
$\mathcal{X}_d, \dot{\mathcal{X}}_d, \ddot{\mathcal{X}}_d$
$\mathcal{F}_{ff} = \hat{M}\ddot{\mathcal{X}}_d + \hat{C}\dot{\mathcal{X}}_d + \hat{G}$
J^{-T}
τ_{ff}
$\mathcal{F}_d$
$\mathcal{X}_d$
Inv. K.
q_d
e_q
$K_d s + K_p$
+
+
−
+
τ
τ_{pd}
Parallel robot
q

Figure 25 – *Feed forward actuator force added to the decentralized PD controller in joint space*

The feedforward term is generated from the dynamic formulation of the manipulator. The desired trajectory in the task space $\boldsymbol{\mathcal{X}}_d$ and its derivatives $\dot{\boldsymbol{\mathcal{X}}}_d$, $\ddot{\boldsymbol{\mathcal{X}}}_d$ are thus required.

In practice, exact knowledge of dynamic matrices are not available, and therefore, estimates of these matrices are used in this derivation denoted by $\hat{\boldsymbol{M}}$, $\hat{\boldsymbol{C}}$ and $\hat{\boldsymbol{G}}$.

The information required to generate the feedforward actuator force $\boldsymbol{\tau}_{ff}$ is usually available beforehand, and in such a case, the feedforward term corresponding to a given trajectory can be **determined off-line**, while the computation of the decentralized feedback term would be executed online.

If complete information of the dynamic matrices is available, and if we assume that the system is performing well, meaning that $\boldsymbol{\mathcal{X}}(t) \simeq \boldsymbol{\mathcal{X}}_d(t)$ and $\dot{\boldsymbol{\mathcal{X}}}(t) \simeq \dot{\boldsymbol{\mathcal{X}}}_d(t)$, we can write the closed loop dynamics as follow:

$$\boldsymbol{M}_q \ddot{\boldsymbol{e}}_q + \boldsymbol{K}_d \dot{\boldsymbol{e}}_q + \boldsymbol{K}_p \boldsymbol{e}_q = \boldsymbol{\tau}_d$$

The error dynamics satisfy a set of second-order differential equations in the presence of disturbance. Therefore, by choosing appropriate gains of the PD controller, the transient and steady-state performance of the tracking error can be suitably designed.
Note that except for the mass matrix, the error dynamics terms are all **configuration independent**, and therefore, it is much easier to tune the PD controller gains to work well in the whole workspace of the robot in such a structure.

However, this method suffers from a number of **limitations** in practice. The most important limitation is the **stringent assumption of the complete information requirement of dynamics matrices**. Furthermore, even is all the assumption hold, because of the configuration dependence of the mass matrix, the error dynamics is still not completely decoupled. This means that correction in one component may be considered as a disturbance effect to the other components. To overcome these limitations, the inverse dynamic approach has been developed and is given in the following section.

d Inverse Dynamics Control

As seen in the previous section, the tracking performance of a decentralized PD controller implemented in the joint space is not uniform at different configurations. To compensate for such effects, a feedforward torque is added to the structure of the controller, by which the shortcomings of the decentralized controller is partially remedied. However, the closed-loop performance still faces a number of limitations, which cannot be completely remedied because of the inherent conditions on feedforward structure of that proposed controller. To overcome these limitations, in this section, a control technique based on **inverse dynamic feedback** of the manipulator in the joint space is presented.

Inverse Dynamics Control

In the inverse dynamics control (IDC) strategy, the **nonlinear dynamics of the model is used to add a corrective term to the decentralized PD controller**. By this means, the **nonlinear and coupling characteristics** of robotic manipulator is significantly **attenuated**, and therefore, the performance of linear controller is significantly improved.

The general structure of inverse dynamics control applied to a parallel manipulator in the joint space is depicted in Figure 26.
A corrective torque $\boldsymbol{\tau}_{fl}$ is added in a **feedback** structure to the closed-loop system, which is calculated from the Coriolis and Centrifugal matrix, and the gravity vector of the manipulator dynamic formulation in the joint space. Furthermore, the mass matrix is acting in the **forward path**, in addition to the desired trajectory acceleration $\ddot{\boldsymbol{q}}_q$. Note that to generate this term, the **dynamic formulation** of the robot, and its **kinematic and dynamic parameters are needed**. In practice, exact knowledge of dynamic matrices are not available, and there estimates are used.

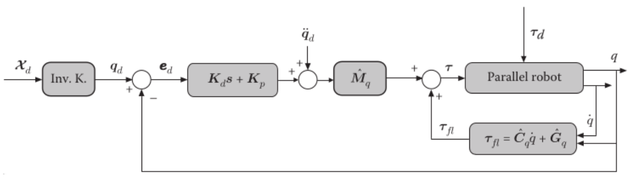
$\boldsymbol{\mathcal{X}}_d$
Inv. K.
q_d
$\boldsymbol{e}_d$
$\boldsymbol{K}_d s + \boldsymbol{K}_p$
$\ddot{q}_d$
$\hat{M}_q$
τ
τ_d
Parallel robot
q
$\dot{q}$
τ_{fl}
$\tau_{fl} = \hat{C}_q \dot{q} + \hat{G}_q$
+
−

Figure 26 – *General configuration of inverse dynamics control implemented in joint space*

The controller output torque applied to the manipulator may be calculated by:

$$\boldsymbol{\tau} = \hat{\boldsymbol{M}}_q \boldsymbol{a}_q + \boldsymbol{\tau}_{fl} \tag{113a}$$

$$\boldsymbol{\tau}_{fl} = \hat{\boldsymbol{C}}_q \dot{\boldsymbol{q}} + \hat{\boldsymbol{G}}_q \tag{113b}$$

$$\boldsymbol{a}_q = \ddot{\boldsymbol{q}}_d + \boldsymbol{K}_d \dot{\boldsymbol{e}}_q + \boldsymbol{K}_p \boldsymbol{e}_q \tag{113c}$$

If the knowledge of dynamic matrices is complete, the closed-loop dynamic formulation is simplified to:

$$\hat{\boldsymbol{M}}_q \left(\ddot{\boldsymbol{e}}_q + \boldsymbol{K}_d \dot{\boldsymbol{e}}_q + \boldsymbol{K}_p \boldsymbol{e}_q \right) + \boldsymbol{\tau}_d = 0$$

This equation implies that if there exist complete knowledge of the dynamic matrices, the tracking error dynamic equation satisfies a set of second-order systems in the presence of disturbance. Consider the case where no disturbance wrench is applied to the manipulator, as the mass matrix $\boldsymbol{M}_q$ is positive definite at all non-singular configurations, it can be inverted, and the error dynamics simplifies to:

$$\ddot{\boldsymbol{e}}_q + \boldsymbol{K}_d \dot{\boldsymbol{e}}_q + \boldsymbol{K}_p \boldsymbol{e}_q = 0$$

This control technique is very popular in practice because of the fact that it can significantly **linearize** and **decouple** the dynamic formulation of the closed-loop system for error dynamics components. Furthermore, the error dynamic terms are all **configuration independent**, and therefore, it is **much easier to tune the PD controller gains** to perform well in the whole workspace of the robot.

However, note that for a good performance, an **accurate model of the system is required**, and the overall procedure is **not robust** to model uncertainties.

6.6 Summary of Motion Control Techniques

In this section, a number of control techniques have been developed for parallel robots. Based on the dynamic formulation given in Section 5, many **model-based** control techniques have been developed for implementation in the task space as well as in the joint space. These control techniques are presented from the simplest form of decentralized PD control to more advanced robust and adaptive inverse dynamics control.
A summary of these techniques is given below.

Dynamic Formulations The dynamic formulation of a parallel robot may be directly represented as a function of motion variable $\boldsymbol{\mathcal{X}}$ in the task space as follows:

$$\boldsymbol{M}(\boldsymbol{\mathcal{X}})\ddot{\boldsymbol{\mathcal{X}}} + \boldsymbol{C}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \boldsymbol{G}(\boldsymbol{\mathcal{X}}) = \boldsymbol{\mathcal{F}} + \boldsymbol{\mathcal{F}}_d$$

The dynamic formulation may be represented as a function of actuator motion variable $\boldsymbol{q}$ as

$$\boldsymbol{M}_q \ddot{\boldsymbol{q}} + \boldsymbol{C}_q \dot{\boldsymbol{q}} + \boldsymbol{G}_q = \boldsymbol{\tau} + \boldsymbol{\tau}_d$$

in which these two formulations are closely related to each other by the following relations:

$$\begin{aligned}
\boldsymbol{M}_q &= \boldsymbol{J}^{-T} \boldsymbol{M} \boldsymbol{J}^{-1} \\
\boldsymbol{C}_q &= \boldsymbol{J}^{-T} \left(\boldsymbol{C} - \boldsymbol{M} \boldsymbol{J}^{-1} \dot{\boldsymbol{J}} \right) \boldsymbol{J}^{-1} \\
\boldsymbol{D}_q &= \boldsymbol{J}^{-T} \boldsymbol{G} \\
\boldsymbol{\tau}_q &= \boldsymbol{J}^{-T} \boldsymbol{\mathcal{F}}
\end{aligned}$$

Decentralized PD Control The simplest controller for a parallel robot can be considered as a decentralized PD controller being implemented individually on each error component. If such a structure is implemented in the task space, the control effort is calculated by

$$\boldsymbol{\mathcal{F}} = \boldsymbol{K}_d \dot{\boldsymbol{e}}_x + \boldsymbol{K}_p \boldsymbol{e}_x$$

and the actuator effort can be generally determined through a force distribution scheme.

For a completely parallel manipulator, the actuator forces can be generated by $\boldsymbol{\tau} = \boldsymbol{J}^{-T}\boldsymbol{\mathcal{F}}$ at non-singular configurations.
Decentralized PD control can be directly implemented in the joint space by the following equation:

$$\boldsymbol{\tau} = \boldsymbol{K}_d\dot{\boldsymbol{e}}_q + \boldsymbol{K}_p\boldsymbol{e}_q$$

Feed Forward Control The reduce the performance limitations of simple PD control, the control effort may be enforced with a feed forward wrench given by

$$\boldsymbol{\mathcal{F}} = \boldsymbol{\mathcal{F}}_{pd} + \boldsymbol{\mathcal{F}}_{ff}$$

in which

$$\begin{aligned}\boldsymbol{\mathcal{F}}_{ff} &= \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x \\ &\quad + \hat{\boldsymbol{M}}(\boldsymbol{\mathcal{X}}_d)\ddot{\boldsymbol{\mathcal{X}}}_d + \hat{\boldsymbol{C}}(\boldsymbol{\mathcal{X}}_d, \dot{\boldsymbol{\mathcal{X}}}_d)\dot{\boldsymbol{\mathcal{X}}}_d + \hat{\boldsymbol{G}}(\boldsymbol{\mathcal{X}}_d)\end{aligned}$$

where $\hat{\boldsymbol{M}}$, $\hat{\boldsymbol{C}}$ and $\hat{\boldsymbol{G}}$ are estimation of the dynamic matrices.
This controller can be implemented in joint space as follows

$$\begin{aligned}\boldsymbol{\tau} &= \boldsymbol{\tau}_{pd} + \boldsymbol{\tau}_{ff} \\ &= \boldsymbol{K}_d\dot{\boldsymbol{e}}_q + \boldsymbol{K}_p\boldsymbol{e}_q + \boldsymbol{J}^{-T}\boldsymbol{\mathcal{F}}_{ff}\end{aligned}$$

Inverse Dynamics Control In the inverse dynamics control, the nonlinear dynamics of the model is used to add a corrective term to the decentralized PD controller. If such a structure is implemented in the task space, the control effort is calculated by

$$\begin{aligned}\boldsymbol{\mathcal{F}} &= \hat{\boldsymbol{M}}(\boldsymbol{\mathcal{X}})\boldsymbol{a} + \hat{\boldsymbol{C}}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \hat{\boldsymbol{G}}(\boldsymbol{\mathcal{X}}) \\ \boldsymbol{a} &= \ddot{\boldsymbol{\mathcal{X}}}_d + \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x\end{aligned}$$

In general, the tracking error dynamics can be represented by

$$\ddot{\boldsymbol{e}}_x + \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x + \hat{\boldsymbol{M}}^{-1}\left[\tilde{\boldsymbol{M}}\ddot{\boldsymbol{\mathcal{X}}} + \tilde{\boldsymbol{C}}\dot{\boldsymbol{\mathcal{X}}} + \tilde{\boldsymbol{G}} + \boldsymbol{\mathcal{F}}_d\right] = 0$$

This controller can be implemented in the joint space as follows:

$$\begin{aligned}\boldsymbol{\tau} &= \hat{\boldsymbol{M}}_q\boldsymbol{a}_q + \hat{\boldsymbol{C}}_q\dot{\boldsymbol{q}} + \hat{\boldsymbol{G}}_q \\ \boldsymbol{a}_q &= \ddot{\boldsymbol{q}}_d + \boldsymbol{K}_d\dot{\boldsymbol{e}}_q + \boldsymbol{K}_p\boldsymbol{e}_q\end{aligned}$$

by which the tracking error dynamics is summarized as

$$\ddot{\boldsymbol{e}}_q + \boldsymbol{K}_d\dot{\boldsymbol{e}}_q + \boldsymbol{K}_p\boldsymbol{e}_q + \hat{\boldsymbol{M}}_q^{-1}\left[\tilde{\boldsymbol{M}}_q\ddot{\boldsymbol{q}} + \tilde{\boldsymbol{C}}_q\dot{\boldsymbol{q}} + \tilde{\boldsymbol{G}}_q + \boldsymbol{\tau}_d\right] = 0$$

Partial Linearization IDC To reduce the computational cost of the inverse dynamic control, it is possible to use partial linearization of dynamic formulation, just by gravity compensation, while keeping asymptotic tracking stability of the closed-loop system. In which a case, the control input wrench in the task space is simplified to

$$\boldsymbol{\mathcal{F}} = \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x + \hat{\boldsymbol{G}}(\boldsymbol{\mathcal{X}})$$

The following Lyapunov function may be used to analyze the stability of tracking dynamics of the closed-loop system:

$$\dot{V} = \dot{\boldsymbol{\mathcal{X}}}^T\boldsymbol{M}\ddot{\boldsymbol{\mathcal{X}}} + \frac{1}{2}\dot{\boldsymbol{\mathcal{X}}}^T\dot{\boldsymbol{M}}\dot{\boldsymbol{\mathcal{X}}} + \boldsymbol{e}_x^T\boldsymbol{K}_p\boldsymbol{e}_x$$

Stability analysis of the closed-loop system in this case reveals the fact that this simplified version of inverse dynamics control can lead to asymptotic tracking for constant desired trajectories.

Robust Inverse Dynamics Control To accommodate modeling uncertainties in inverse dynamic control, the following robust control scheme in the task space is developed:

$$\begin{aligned}\boldsymbol{\mathcal{F}} &= \hat{\boldsymbol{M}}(\boldsymbol{\mathcal{X}})\boldsymbol{a}_r + \hat{\boldsymbol{C}}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \hat{\boldsymbol{G}}(\boldsymbol{\mathcal{X}}) \\ \boldsymbol{a}_r &= \ddot{\boldsymbol{\mathcal{X}}}_d + \boldsymbol{K}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_p\boldsymbol{e}_x + \boldsymbol{\delta}_a\end{aligned}$$

in which the robustifying corrective term $\boldsymbol{\delta}_a$ is found through a Lyapunov stability analysis of tracking error dynamics. The tracking error dynamics can be represented by the following linear and nonlinear components:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ -\boldsymbol{K}_p & -\boldsymbol{K}_d \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{I} \end{bmatrix}$$

The corrective term $\boldsymbol{\delta}_a$ can be found as

$$\boldsymbol{\delta}_a = \begin{cases} -\rho\frac{v}{\|v\|} & \text{if} \|v\| > \epsilon \\ -\rho\frac{v}{\epsilon} & \text{if} \|v\| \le \epsilon \end{cases}$$

in which v is defined by $v = \boldsymbol{B}^T\boldsymbol{P}\boldsymbol{\epsilon}$, where $\boldsymbol{P}$ is the solution to the matrix Lyapunov equation and ϵ is a smoothing threshold. It is shown that by adding this corrective term to the regular inverse dynamics control, the closed-loop system achieves uniform ultimate bounded tracking errors.

Adaptive Inverse Dynamics Control In the adaptive version of the inverse dynamics control, full feedback linearization is considered through adaptive update of dynamic formulation matrices. The error dynamics in this case is

$$\dot{\boldsymbol{\epsilon}} = \boldsymbol{A}\boldsymbol{\epsilon} + \boldsymbol{B}\boldsymbol{\Phi}\tilde{\boldsymbol{\theta}}$$

in which

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \\ -bmK_p & -\boldsymbol{K}_d \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{I} \end{bmatrix}$$

$$\boldsymbol{\Phi} = \hat{\boldsymbol{M}}^{-1}\boldsymbol{\Upsilon}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}}, \ddot{\boldsymbol{\mathcal{X}}})$$

Based on the Lyapunov stability analysis, by using the following Lyapunov function

$$V = \boldsymbol{\epsilon}^T\boldsymbol{P}\boldsymbol{\epsilon} + \tilde{\boldsymbol{\theta}}^T\boldsymbol{\Gamma}\boldsymbol{\theta}$$

the following parameter adaptation law is derived for updates

$$\dot{\hat{\boldsymbol{\theta}}} = -\boldsymbol{\Gamma}^{-1}\boldsymbol{\Phi}^T\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{\epsilon}$$

By this means, the closed-loop system achieves asymptotic tracking performance, while the parameter estimation errors remain bounded.

6.7 Motion Control of the Stewart-Gough Platform

a Control in the Task space

For the Stewart-Gough platform, the motion variable in the task space is a six-dimensional vector

$$\boldsymbol{\mathcal{X}} = \begin{bmatrix} \boldsymbol{x}_p \\ \boldsymbol{\theta} \end{bmatrix}$$

with:

- $\boldsymbol{x}_p = [x_p \ y_p \ z_p]^T$ is the position vector of the motion platform center of mass
- $\boldsymbol{\theta} = \theta [s_x \ s_y \ s_z]^T = [\theta_x \ \theta_y \ \theta_z]^T$ is the moving platform orientation expressed by screw coordinates

Therefore, the tracking error is defined as $\boldsymbol{e} = [e_x \ e_y \ e_z \ e_{\theta_x} \ e_{\theta_y} \ e_{\theta_z}]^T$.
The decentralized controller consists of six disjoint proportional derivative controllers acting on each error component and is denoted by $\boldsymbol{K}_d s + \boldsymbol{K}_p$.

The output of the controller is denoted by $\boldsymbol{\mathcal{F}} = [F_x \ F_y \ F_z \ \tau_x \ \tau_y \ \tau_z]^T$. Note that since the output of the controller is defined in the task space, each wrench component directly manipulates the corresponding tracking error component, and therefore, the overall tracking performance of the manipulator is suitable is high controller gains are used.

In practice, the calculated output wrench is transformed into actuator forces through the force distribution block corresponding to the inverse of Jacobian transpose.

b Control in the Joint space

The joint variable $\boldsymbol{q}(t)$ is a six-dimensional vector consisting of the limb lengths denoted by $\boldsymbol{q} = [l_1 \ l_2 \ l_3 \ l_4 \ l_5 \ l_6]^T$. Therefore, the tracking error is defined as $\boldsymbol{e}_q = \boldsymbol{q}_d - \boldsymbol{q}$, in which is the desired motion variable in the joint space $\boldsymbol{q}_d$ is determined by the solution of inverse kinematics, and $\boldsymbol{q}$ is given by direct measurement of the limb lengths.

The decentralized controller, therefore, consists of six disjoint PD controllers acting on each error component. The output of the controller directly generates the actuator torques denoted by τ.

In simulation, it is observe that the orientation error significantly increase in the joint space control scheme. The main reason is that the controller gains directly penalize the position error of the limb lengths, and not the orientation errors, and therefore, there is no direct controller action to be suitably tuned to reduce the orientation error.

Comparing the closed-loop performance of the PD controllers designed in the joint space to those designed in the task space, it can be concluded that **tuning of the PD gains for a suitable performance is much easier in task space designs**. Furthermore, a very small error signature in the joint space may be accumulated to produce relatively larger tracking errors in the task space. Hence, it is recommended to design and implement controllers in the task space, if the required motion variables can be directly measured or the forward kinematic solution can be calculated in an online routine.

7 Force Control

7.1 Introduction

In many applications, it may occur that the robot moving platform is in contact with a **stiff** environment and **specific interacting wrench is required**. In such applications, the contact wrench describes the state of interaction more effectively than the position and orientation of the moving platform.

Force Control

The problem of force control can be described as to derive the actuator forces required to generate a prescribed desired wrench at the manipulator moving platform, when the manipulator is carrying out its desired motion.

This problem and its extents are treated in the **force control** algorithms described in this chapter. A force control strategy is one that modifies position trajectories based on the sensed wrench, or force-motion relations.

If a pure **motion control** scheme is used for manipulator, in case it contacts an environment, the robot does not sense the presence of the environment, and its driving forces become harshly high to reduce the tracking errors. In such a case, the robot may break the object it is in contact or will break its internal structure. Additional sensors should be included in the manipulator in order for it to be able to feel the interaction and to control the interacting forces. Different wrench sensors are developed for such applications, and it is possible to use joint torque or link force measurement units to determine the projection of the interacting forces in the joint space.

The use of wrench sensors either in the task space or in the joint space open horizons to use different force control topologies for the manipulators. Using such sensors does not imply that regular motion sensors used in motion control schemes are not necessary. The use of motion sensors and the usual corresponding control topologies are usually necessary, since the motion of the manipulator is one of the outputs to be controlled. Depending on the type and configuration of the wrench sensors, different force control topologies are developed.

7.2 Controller Topology

For a force control scheme, the desired interacting wrench of the moving platform and the environment may be of interest. This quantity may be denoted by $\boldsymbol{\mathcal{F}}_d$, which has the same dimension and structure of the manipulator wrench $\boldsymbol{\mathcal{F}}$. To carry out such a control task in a closed-loop structure, it is necessary to **measure the output wrench** of the manipulator through an instrumentation system.

Although there are many commercial six-degrees-of-freedom wrench sensors available in the market, they are usually more expensive than single joint force measurement units. Another alternative for force measurement is **direct measurement of the actuator forces**. Many commercial linear actuators are available in the market in which **embedded force measurement** is considered in their design. Therefore, it might be preferable in some applications to use direct actuator force measurements to carry out the feedback control.

a Cascade Control

In a general force control scheme, the **prime objective** is tracking of the interacting wrench between the moving platform and the environment. However, note that the motion control of the robot when the robot is in interaction with the environment is also another **less-important objective** and when the contact of the robot moving platform is released, motion control becomes the prime objective.

Cascade Control

To follow **two objectives** with different properties in one control system, usually a **hierarchy** of two feedback loops is used in practice. This kind of control topology is called **cascade control**, which is used when there are **several measurements and one prime control variable**. Cascade control is implemented by **nesting** the control loops, as shown in Figure 27. The output control loop is called the **primary loop**, while the inner loop is called the secondary loop and is used to fulfill a secondary objective in the closed-loop system.

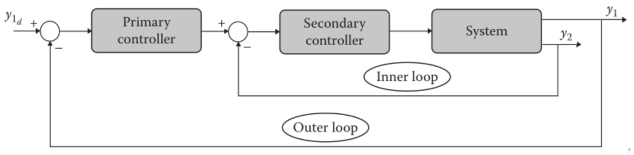

Figure 27 – *Block diagram of a closed-loop system with cascade control*

The measured variables are here the motion and interacting wrench that may be measured in the task space or in the joint space, and therefore, different control topologies may be advised for each set of measurement variables.

To improve the performance of the control system for a particular objective, it is important to **choose the right variables for internal and external feedback loops**, and to **design suitable controllers for each feedback system**. Although these differ in different topologies described in the following sections, some **general rules** are applied to design a well performing cascade control system.
A general idea in cascade control design is the **ideal case**, in which the inner loop is designed so tight that the secondary (inner) loop behaves as a **perfect servo**, and responds very quickly to the internal control command. This idea is effectively used in many applications, wherein a nearly-perfect actuator to respond to the requested commands is designed by using an inner control feedback.

Design criteria - Inner loop

The design criteria for the inner loop is to have a high control gain such that the time response of the secondary variable is at least 5 times more than that of the primary variable, and such that it can overcome the effect of **disturbances** and **unmodelled dynamics** in the **internal feedback structure**.

It is also necessary to have a well-defined relation between the primary and secondary variables, to have harmony in the objectives followed in the primary and secondary loops.

b Force Feedback in Outer Loop

Consider the force control schemes, in which **force tracking is the prime objective**. In such a case, it is advised that the outer loop of cascade control structure is constructed by wrench feedback, while the inner loop is based on position feedback. Since different types of measurement units may be used in parallel robots, different control topologies may be constructed to implement such a cascade structure.

Consider first the cascade control topology shown in Figure 28 in which the measured variables are both in the **task space**. The inner loop is constructed by position feedback while the outer loop is based on force feedback. As seen in Figure 28, the force controller block is fed to the motion controller, and this might be seen as the **generated desired motion trajectory for the inner loop**.
The output of motion controller is also designed in the task space, and to convert it to implementable actuator force $\boldsymbol{\tau}$, the force distribution block is considered in this topology.

Other alternatives for force control topology may be suggested based on the variations of position and force

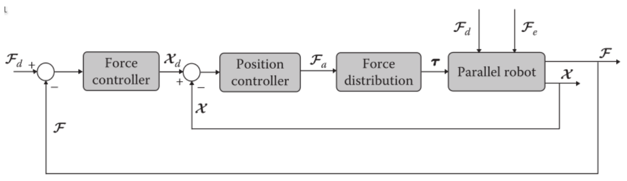

Figure 28 – *Cascade topology of force feedback control: position in inner loop and force in outer loop. Moving platform wrench $\boldsymbol{\mathcal{F}}$ and motion variable $\boldsymbol{\mathcal{X}}$ are measured in the task space*

measurements. If the force is measured in the joint space, the topology suggested in Figure 29 can be used. In this topology, the measured actuator force vector $\boldsymbol{\tau}$ is mapped into its corresponding wrench in the task space by the Jacobian transpose mapping $\boldsymbol{\mathcal{F}} = \boldsymbol{J}^T\boldsymbol{\tau}$.

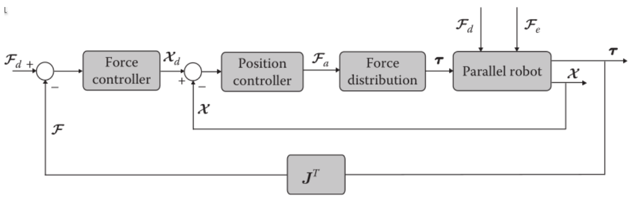

Figure 29 – *Cascade topology of force feedback control: position in inner loop and force in outer loop. Actuator forces $\boldsymbol{\tau}$ and motion variable $\boldsymbol{\mathcal{X}}$ are measured*

Consider the case where the force and motion variables are both measured in the **joint space**. Figure 30 suggests the force control topology in the joint space, in which the inner loop is based on measured motion variable in the joint space, and the outer loop uses the measured actuator force vector. In this topology, it is advised that the force controller is designed in the **task** space, and the Jacobian transpose mapping is used to project the measured actuator force vector into its corresponding wrench in the task space. However, as the inner loop is constructed in the joint space, the desired motion variable $\boldsymbol{\mathcal{X}}_d$ is mapped into joint space using **inverse kinematic** solution.
Therefore, the structure and characteristics of the position controller in this topology is totally different from that given in the first two topologies.

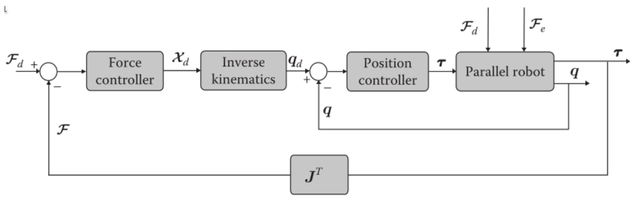

Figure 30 – *Cascade topology of force feedback control: position in inner loop and force in outer loop. Actuator forces $\boldsymbol{\tau}$ and joint motion variable $\boldsymbol{q}$ are measured in the joint space*

c Force Feedback in Inner Loop

Consider the force control scheme in which the **motion-force relation is the prime objective**. In such a case, force tracking is not the primary objective, and it is advised that the outer loop of cascade control structure consists of a motion control feedback.
Since different type of measurement units may be used in parallel robots, different control topologies may be constructed to implement such cascade controllers.

Figure 31 illustrates the cascade control topology for the system in which the measured variables are both in the task space ($\boldsymbol{\mathcal{F}}$ and $\boldsymbol{\mathcal{X}}$). The inner loop is loop is constructed by force feedback while the outer loop is based on position feedback. By this means, when the manipulator is not in contact with a stiff environment, position tracking is guaranteed through the primary controller. However, when there is interacting wrench $\boldsymbol{\mathcal{F}}_e$ applied to the moving platform, this structure controls the force-motion relation. This configuration may be seen as if the **outer loop generates a desired force trajectory for the inner loop**.

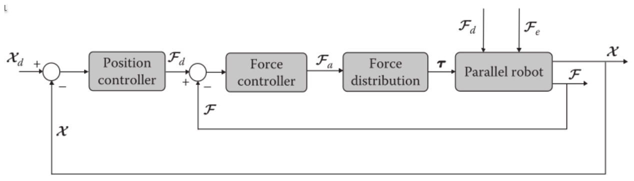

Figure 31 – *Cascade topology of force feedback control: force in inner loop and position in outer loop. Moving platform wrench $\boldsymbol{\mathcal{F}}$ and motion variable $\boldsymbol{\mathcal{X}}$ are measured in the task space*

Other alternatives for control topology may be suggested based on the variations of position and force measurements. If the force is measured in the joint space, control topology shown in Figure 32 can be used. In such case, the Jacobian transpose is used to map the actuator force to its corresponding wrench in the task space.

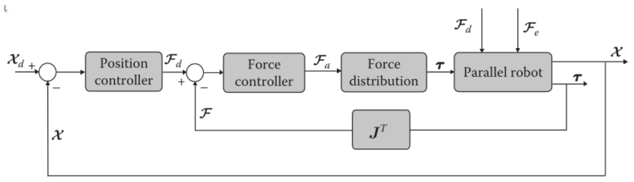

Figure 32 – *Cascade topology of force feedback control: force in inner loop and position in outer loop. Actuator forces $\boldsymbol{\tau}$ and motion variable $\boldsymbol{\mathcal{X}}$ are measured*

If the force and motion variables are both measured in the **joint** space, the control topology shown in Figure 33 is suggested. The inner loop is based on the measured actuator force vector in the joint space $\boldsymbol{\tau}$, and the outer loop is based on the measured actuated joint position vector $\boldsymbol{q}$. In this topology, the desired motion in the task space is mapped into the joint space using **inverse kinematic** solution, and **both the position and force feedback controllers are designed in the joint space**. Thus, independent controllers for each joint may be suitable for this topology.

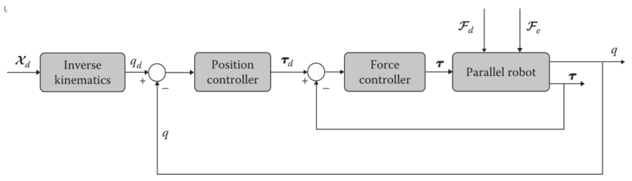

Figure 33 – *Cascade topology of force feedback control: force in inner loop and position in outer loop. Actuator forces $\boldsymbol{\tau}$ and joint motion variable $\boldsymbol{q}$ are measured in the joint space*

7.3 Stiffness Control

a Single-Degree-of-Freedom Stiffness Control

b General Stiffness Control

c Stiffness Contorl of the Stewart-Gough Platform

7.4 Direct Force Control

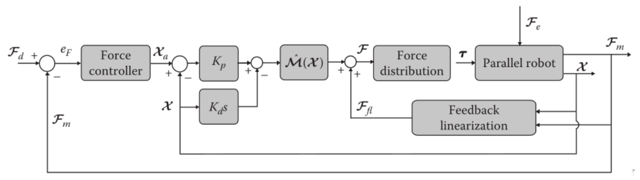

Figure 34 – *Direct force control scheme, force feedback in the outer loop and motion feedback in the inner loop*

7.5 Impedance Control

For the stiffness control and direct force control schemes, it is observed that when the manipulator-moving platform is in contact with a stiff environment, the motion variable $\boldsymbol{\mathcal{X}}$ and the interacting force variable $\boldsymbol{\mathcal{F}}$ are two dynamically **dependent** quantities.

In stiffness control, it is aimed to adjust the **static** relation between these two quantities. In this scheme, no force measurement is required, however, careful design on the desired motion trajectory and PD controller gains is needed to tune the stiffness property of the interaction at steady stage.

In force control schemes, on the other hand, the force tracking is the prime objective, and force measurement is a stringent requirement to implement such schemes.

The main reason that the motion and force variables are not being gable to be controlled independently is that for an n-degrees-of-freedom manipulator, only n-independent control inputs are available, and therefore, only n-independent variables can be controlled, while the force and motion quantities count to $2n$ independent variables.

Impedance Control

The key idea behind **impedance control** schemes, is to **tune the dynamic relation between the force and the motion variables**, and not a hierarchy of tracking objectives in force and in position variables. In this scheme, contrary to stiffness control schemes, **both force and position variables are measured** and used in the control structure.

The definition of mechanical **impedance** is given in an analogy of the well-known electrical impedance definition as the **relationship between the effort and flow variables**. Since this relation can be well determined in the frequency domain, the dynamical relation of force and motion variable may be represented by mechanical impedance. Impedance control schemes provide control topology to tune the mechanical impedance of a system to a desired value. By this means, the force and the motion variables are not controlled independently, or in a hierarchy, but their dynamic relation represented by mechanical impedance is suitably controlled.

a Impedance

Impedance was first defined in electrical networks as the measure of the opposition that an electrical circuit presents to the passage of a current when a voltage is applied. To **generalize** the impedance definition to other disciplines, voltage is generalized to the **effort** and current is generalized to the **flow**.

Impedance is a **complex** function defined as the ratio of the Laplace transform of the effort to the Laplace transform of the flow.

Impedance is usually denoted by $\boldsymbol{Z}(s)$ and it may be represented by writing its magnitude and phase in the form of $|\boldsymbol{Z}(s)|$ and $\angle\boldsymbol{Z}(s)$. The magnitude of the complex impedance $|\boldsymbol{Z}|$ is the ratio of the effort amplitude to that of the flow, while the phase $\angle\boldsymbol{Z}$ is the phase shift by which the flow is ahead of the effort.

Mechanical Impedance - Definition

Mechanical Impedance is defined as the ratio of the Laplace transform of the mechanical effort to the Laplace transform of the mechanical flow:

$$\boldsymbol{Z}(s) = \frac{\boldsymbol{F}(s)}{\boldsymbol{v}(s)} \tag{114}$$

in which effort in mechanical systems is represented by force $\boldsymbol{F}$ and flow is represented by velocity $\boldsymbol{v}$.

Note that this definition is given for a single-degree-of-freedom motion system. The motion can be generalized to angular motion, in which the effort is represented by torque, while the flow is represented by angular velocity. Furthermore, the impedance may be generalized to multiple-degrees-of-freedom system, in which for a general spatial motion effort is represented by a wrench $\boldsymbol{\mathcal{F}}$, while flow is represented by motion twist $\dot{\boldsymbol{\mathcal{X}}}$.

Nevertheless, note that Laplace transform is only applicable for **linear time invariant** systems, and for a parallel manipulator the dynamic formulation of which is nonlinear, the concept of mechanical impedance may be extended to the differential equation relating the mechanical wrench $\boldsymbol{\mathcal{F}}$ to motion twist $\dot{\boldsymbol{\mathcal{X}}}$.

Example - RLC circuit

Consider an RLC circuit depicted in Figure 35. The differential equation relating voltage v to the current i is given by

$$v = L\frac{di}{dt} + Ri + \int_0^t \frac{1}{C} i(\tau)d\tau$$

in which L denote the inductance, R the resistance and C the capacitance.
The impedance of the system may be found from the Laplace transform of the above equation:

$$Z(s) = \frac{v(s)}{i(s)} = Ls + R + \frac{1}{Cs} \tag{115}$$

Example - Mass-Spring-Damper

Consider the mass-spring-damper system depicted in Figure 35. The governing dynamic formulation for this system is given by

$$m\ddot{x} + c\dot{x} + kx = f$$

in which m denote the body mass, c the damper viscous coefficient and k the spring stiffness.
The impedance of the system may be found from the Laplace transform of the above equation:

$$Z(s) = \frac{f(s)}{v(s)} = ms + c + \frac{k}{s} \tag{116}$$

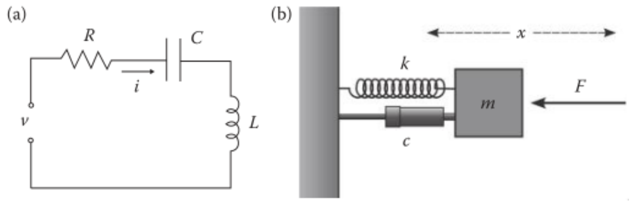

Figure 35 – *Analogy of electrical impedance in (a) an electrical RLC circuit to (b) a mechanical mass-spring-damper system*

As inferred from the above two examples, although the physical nature of the system may differ from each other, they may be represented by similar impedances. From this analogy, a terminology for impedance is introduced.

Impedance - Terminology

An impedance $\boldsymbol{Z}(s)$ is called

- **Inductive** if $|\boldsymbol{Z}(0)| = 0$
- **Resistive** if $|\boldsymbol{Z}(0)| = R$
- **Capacitive** if $\lim_{s\to 0} |\boldsymbol{K}(s)| = \infty$

Hence, for the mechanical system represented in Figure 35:

- mass represents inductive impedance
- viscous friction represents resistive impedance
- spring stiffness represents capacitive impedance

The environments that a robot interacts with may be represented by these classified impedance components. A very stiff environment may be represented by high-capacitive impedance models, whereas an environment with high structural damping may be represented by high-resitive impedance.

b Impedance Control Concept

The key idea behind impedance control schemes is to tune the dynamic relation between force and motion variables. Impedance control schemes provide control topology to tune the mechanical impedance of a system toward a desired impedance.
A desired impedance could be adjusted by desired inductive, resistive and capacitive impedances, which forms a desired linear impedance for the closed-loop system as follows:

$$\boldsymbol{Z}_d(s) = \boldsymbol{M}_d s + \boldsymbol{C}_d + \frac{1}{s}\boldsymbol{K}_d$$

where $\boldsymbol{Z}_d$ denotes the desired impedance of the closed-loop system, which is composed of the desired inductive impedance $\boldsymbol{M}_d$, desired resistive impedance $\boldsymbol{C}_d$ and desired capacitive impedance $\boldsymbol{K}_d$. Impedance control structures may be used to tune the closed-loop impedance of the system suitably to follow such a desired impedance.

c Impedance Control Structure

Consider a parallel robot with multiple-degrees-of-freedom interacting with a stiff environment. In such a case, the motion of the robot end effector is represented by the motion vector $\boldsymbol{\mathcal{X}}$, and the interacting wrench applied to the robot end effector is denoted by $\boldsymbol{\mathcal{F}}_e$. It is considered that the interacting wrench is measured in the task space and is used in the inner force feedback loop. Furthermore, it is considered that the motion variable $\boldsymbol{\mathcal{X}}$ is measured and is used in the outer feedback loop.

In the impedance control scheme, **regulation of the motion-force dynamic relation is the prime objective**, and since force tracking is not the primary objective, it is advised to used a cascade control structure with motion control feedback in the outer loop and force feedback in the inner loop.
Therefore, when the manipulator is not in contact with a stiff environment, position tracking is guaranteed by a primary controller. However, when there is an interacting wrench $\boldsymbol{\mathcal{F}}_e$ applied to the moving platform, this structure may be designed to control the force-motion dynamic relation.

As a possible impedance control scheme, consider the closed-loop system depicted in Figure 36, in which the position feedback is considered in the outer loop, while force feedback is used in the inner loop. This structure is advised when a desired impedance relation between the force and motion variables is required that consists of desired inductive, resistive, and capacitive impedances. As shown in Figure 36, the motion-tracking error is directly determined from motion measurement by $\boldsymbol{e}_x = \boldsymbol{\mathcal{X}}_d - \boldsymbol{\mathcal{X}}$ in the outer loop and the motion controller is designed to satisfy the required impedance.
Moreover, direct force-tracking objective is not assigned in this control scheme, and therefore the desired force trajectory $\boldsymbol{\mathcal{F}}_d$ is absent in this scheme. However, an auxiliary force trajectory $\boldsymbol{\mathcal{F}}_a$ is generated from the motion control law and is used as the reference

for the force tracking. By this means, no prescribed force trajectory is tracked, while the **motion control scheme would advise a force trajectory for the robot to ensure the desired impedance regulation**.

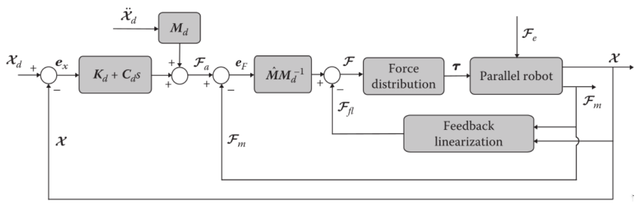

Figure 36 – *Impedance control scheme; motion feedback in the outer loop and force feedback in the inner loop*

The required wrench $\boldsymbol{\mathcal{F}}$ in the impedance control scheme, is based on inverse dynamics control and consists of three main parts. In the inner loop, the force control scheme is based on a feedback linearization part in addition to a mass matrix adjustment, while in the outer loop usually a linear motion controller is considered based on the desired impedance requirements. Although many different impedance structures may be considered as the basis of the control law, in Figure 36, a linear impedance relation between the force and motion variables is generated that consists of desired inductive $\boldsymbol{M}_d$, resistive $\boldsymbol{C}_d$ and capacitive impedances $\boldsymbol{K}_d$.

According to Figure 36, the controller output wrench $\boldsymbol{\mathcal{F}}$, applied to the manipulator may be formulated as

$$\boldsymbol{\mathcal{F}} = \hat{\boldsymbol{M}}\boldsymbol{M}_d^{-1}\boldsymbol{e}_F + \boldsymbol{\mathcal{F}}_{fl}$$

with:

$$\begin{aligned}\boldsymbol{e}_F &= \boldsymbol{\mathcal{F}}_a - \boldsymbol{\mathcal{F}}_m\\ \boldsymbol{\mathcal{F}}_a &= \boldsymbol{M}_d\ddot{\boldsymbol{\mathcal{X}}}_d + \boldsymbol{C}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_d\boldsymbol{e}_x\end{aligned}$$

$\boldsymbol{M}_d$ denotes the desired inductive impedance, $\boldsymbol{C}_d$ the desired resistive impedance and $\boldsymbol{K}_d$ is desired capacitive impedance matrices.
The feedback linearizing term is given by:

$$\boldsymbol{\mathcal{F}}_{fl} = \hat{\boldsymbol{C}}(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})\dot{\boldsymbol{\mathcal{X}}} + \hat{\boldsymbol{G}}(\boldsymbol{\mathcal{X}}) + \boldsymbol{\mathcal{F}}_m$$

If the information on the dynamic matrices is complete, and if the force measurements are noise free ($\boldsymbol{\mathcal{F}}_m = \boldsymbol{\mathcal{F}}_e$), the closed-loop dynamic formulation simplifies to:

$$\boldsymbol{M}_d\ddot{\boldsymbol{e}}_x + \boldsymbol{C}_d\dot{\boldsymbol{e}}_x + \boldsymbol{K}_d\boldsymbol{e}_x = \boldsymbol{\mathcal{F}}_e$$

And thus the closed-loop error dynamic equation satisfies a set of second-order systems with the desired impedance coefficients in relation to the interacting force. In other words, **the control structure guarantees that the force and motion relation follows a desired impedance**. Therefore, by choosing appropriate impedance matrices, the transient performance of the force-motion relation can be shaped so as to have a fast but stable interaction. By this means, what is controlled is the dynamic relation between the force and motion variables, and not directly the position. However, if the robot is moving freely in space and has no interaction with the environment, $\boldsymbol{\mathcal{F}}_e = 0$, the closed-loop system will provide a suitable motion tracking thanks to the motion controller in the outer loop.

The impedance control scheme is very popular in practice, wherein tuning the force and motion relation in a robot manipulator interacting with a stiff environment is the prime objective. However, note that for a good performance, an accurate model of the system is required, and the obtained force and motion dynamics are not robust to modeling uncertainty.